EXTRACTION DE CARACTÉRISTIQUES : CONTOURS MULTISPECTRAUX, CONTOURS DE TEXTURE ET ROUTES

par

Marie-Flavie Auclair Fortier

mémoire présenté au Département de mathématiques et d'informatique
en vue de l'obtention du grade de maître ès sciences (M.Sc.)

FACULTÉ DES SCIENCES
UNIVERSITÉ DE SHERBROOKE

Sherbrooke, Québec, Canada, octobre 1999

Le 5 novembre 99, le jury suivant a accepté ce mémoire dans sa version finale.
date

Président-rapporteur:	M. Marc Frappier Département de mathématiques et d'informatique	_____
Membre:	M. Shengrui Wang Département de mathématiques et d'informatique	_____
Membre:	M. Djemel Ziou Département de mathématiques et d'informatique	_____
Membre:	M. Costas Armenakis Géomatique Canada	_____

SOMMAIRE

Ce mémoire concerne l'extraction de caractéristiques. Nous divisons les caractéristiques en deux types : de bas niveau ou non-nominatives et de haut niveau ou nominatives. Nous proposons dans un premier temps un détecteur original de contours de type Laplacien dans les images multispectrales, basé sur les dérivées d'une fonction de contraste présentée par Cumani. Dans un deuxième temps, nous présentons un second détecteur de contours original pour les images de texture. Ce détecteur est basé sur les différences des moments locaux d'ordre supérieur. Afin de combiner les contours dûs aux différents moments, nous utilisons l'approche multispectrale. Ces deux détecteurs extraient des caractéristiques de bas niveau. Concernant les caractéristiques de haut niveau, nous présentons un détecteur de routes dans les images à haute résolution après avoir étudié les différentes méthodes existantes dans un rapport externe. Ce détecteur utilise l'information contenue dans une base de données de routes, fournie par Géomatique Canada, comme initialisation pour des contours actifs. Ces contours actifs servent à corriger cette même information, imprécise car elle est obtenue par numérisation des cartes routières. Des hypothèses pour les routes manquantes sont générées par un suivi de route débutant du réseau routier connu. Des résultats pour les trois détecteurs sont présentés démontrant la validité des approches utilisées.

REMERCIEMENTS

Je tiens tout d'abord à remercier M. Djemel Ziou, mon directeur de recherche qui m'a fait connaître et apprécier la vision par ordinateur. Je veux particulièrement le remercier de sa disponibilité, de ses conseils judicieux et aussi de son humour.

Je veux ensuite remercier M. Costas Armenakis, mon co-directeur, et Géomatique Canada pour m'avoir fournis un projet intéressant, du financement ainsi que certaines données.

Je tiens à remercier également les autres membres du jury, M. Shengrui Wang et M. Marc Frappier.

Je remercie particulièrement François Deschênes qui a développé le détecteur de jonctions de lignes en vue de l'intégration à notre application de détection de routes et Pierre Poulin qui a programmé l'interface graphique sous Windows pour cette même application.

Je remercie mes parents ainsi que Claude Delisle pour leur support et leurs encouragements tout au long de cette maîtrise. Enfin, je remercie Scott Lawrence et François Deschênes pour les discussions enrichissantes sur la vision et autres sujets, ainsi que toute la bande du 1021 pour avoir rendu plus agréable la fin de ces études.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

INTRODUCTION

L'émulation du système visuel par l'ordinateur est un processus complexe. En effet, le sens de la vision est celui qui peut traiter le plus d'informations parmis les cinq sens humains. La vision par ordinateur implique donc plusieurs étapes. Une des étapes de base est de réduire la quantité d'informations de façon à ne retenir que celles qui sont pertinentes. Cette étape est l'extraction de caractéristiques. Ces caractéristiques sont divisées en deux catégories: de haut niveau et de bas niveau. Les caractéristiques de haut niveau sont des éléments nominatifs, c'est-à-dire qu'elles portent un nom et donc, la plupart du temps, sont rattachées à une application en particulier de la vision artificielle. Nous pouvons penser par exemple à la recherche d'éléments géographiques comme des lacs, des routes ou des terres cultivées dans les images aériennes, ou à la localisation des mains dans une application d'interprétation du langage des signes. Les caractéristiques de bas niveau sont plutôt des éléments non nominatifs. Elles comprennent, entre autres, les contours de différents types (marche, ligne, crête, jonction, ...) dans des images à niveaux de gris, multispectrales ou de texture. Les caractéristiques de bas niveau sont utilisées, combinées avec des informations de contexte, dans la recherche des caractéristiques de haut niveau.

Dans ce mémoire, nous nous intéressons aux deux types de caractéristiques. D'abord, nous recherchons deux caractéristiques de bas niveau, les contours dans les images multispectrales et les contours dans les images de texture. Ensuite, nous recherchons une

caractéristique de haut niveau, c'est-à-dire les routes dans les images aériennes ou satellites à haute résolution.

Les images multispectrales sont des images ayant plusieurs bandes, c'est-à-dire qu'à chaque pixel correspond un vecteur de valeurs. Par exemple, les images couleurs ont trois bandes et les images Landsat-TM en ont sept. La plupart des détecteurs de contours ont été développés pour les images à niveaux de gris (une seule bande) alors que le multispectral peut apporter un supplément d'information très utile. Dans la suite de certains travaux [8, 13, 15] sur le gradient multispectral, nous développons au chapitre 1 un opérateur Laplacien original pour les images multispectrales en dérivant une fonction de contraste présentée par Cumani [8]. Un tel Laplacien peut être utile dans la recherche de certains types de jonctions ou dans la reconstruction 3D par exemple.

Les images de texture sont des images comprenant des régions ne pouvant être décrites uniquement par le niveau de gris moyen. Les frontières de ces régions ne peuvent donc toujours être trouvées par les détecteurs de contours usuels puisqu'ils sont basés sur les différences entre les niveaux de gris moyens. Un des moyens de décrire une région de texture est l'utilisation des moments d'ordre supérieur. Cependant, aucun détecteur de contours de texture existant n'utilise les moments locaux. Nous basons notre détecteur de contours de texture, présenté au chapitre 2, sur les différences de ces moments locaux. Pour obtenir une seule image de contours pour l'ensemble des moments, nous reprennons l'approche multispectrale. En effet, nous fabriquons une image multibande avec, dans chacune des bandes, une image de moments en plus de l'image originale. Nous choisissons cette solution, car la fusion des résultats individuels de chaque moment est un procédé complexe.

En dernier lieu, au chapitre 3, nous présentons un détecteur de routes dans les images à haute résolution. Ce détecteur de routes est développé dans un contexte de mise-à-jour de cartes routières fournies par Géomatique Canada. Comme nous avons accès à

des informations connues (les anciennes cartes), nous faisons face à deux problèmes; la correction des informations existantes et l'ajout des nouvelles routes. Pour la correction des informations, nous utilisons un processus itératif d'optimisation des contours (les contours actifs) nécessitant une initialisation. Cette initialisation nous est fournie par l'information existante. Pour cette étape, nous modifions une approche existante [24] en y ajoutant une détection de jonctions de lignes développée dans notre groupe de recherche [12]. Par la suite, nous générons des hypothèses de nouvelles routes en effectuant un suivi des lignes connectées au réseau existant.

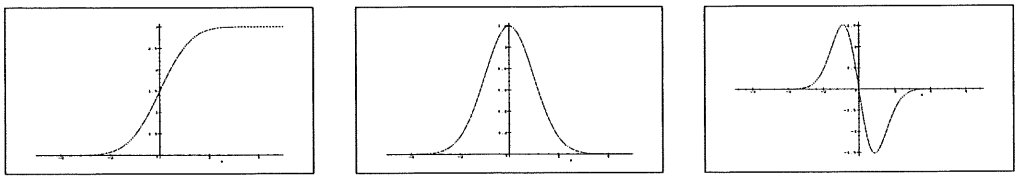
En résumé, le reste du mémoire est organisé comme suit. Le chapitre 1 présente le Laplacien multispectral. Le chapitre 2 porte sur la détection de contours dans les images de texture ainsi que sur le détecteur de contours de texture que nous proposons. Le chapitre 3 présente notre approche pour la détection des routes dans les images à haute résolution. Finalement, nous présentons une conclusion générale à la fin du mémoire. Chacun des chapitres comprend une introduction présentant le problème, le développement du détecteur, une section de résultats et une conclusion. Les chapitres 1 et 2 comprennent de plus une section sur les travaux existants dans les domaines traités par chaque chapitre. Les travaux sur la détection de routes sont présentés dans un rapport de recherche externe au mémoire [3].

CHAPITRE 1

DÉTECTION DE CONTOURS MULTISPECTRAUX

1.1 Introduction

Une image est une projection d'une scène tridimentionnelle dans un plan bidimentionnel. Selon le type de capteur utilisé, une image peut être à niveaux de gris (une bande) ou multispectrale (plusieurs bandes). Dans le cas des images à niveaux de gris (i.e. une fonction de $\mathbb{R}^2 \Rightarrow \mathbb{R}$), des phénomènes physiques, géométriques et photométriques dans la scène, engendrent des variations dans l'image. Ces variations sont appelées contours. Il existe plusieurs types de contours dont les marches, les lignes et les jonctions. Les contours de type marche dans les images à niveaux de gris (figure 1.1a) ont fait l'objet de nombreuses études et plusieurs détecteurs ont été proposés [37]. Une classe importante de ces détecteurs est celle comportant le calcul des dérivées de l'image. Les contours de type marche sont détectés aux maxima de la valeur absolue de la première dérivée (figure 1.1b) ou aux passages par zéro de la seconde dérivée (figure 1.1c). En deux dimensions,



a) b) c)

Figure 1.1 – *(a) Contour de type marche. (b) Première dérivée. (c) Seconde dérivée.*

la première dérivée se traduit approximativement par le gradient et la deuxième par le Laplacien ou la deuxième dérivée directionnelle dans la direction du gradient.

Dans le cas des images multispectrales, c'est-à-dire qu'à chaque point de l'image correspond un vecteur (une fonction de $\mathbb{R}^2 \Rightarrow \mathbb{R}^m$), les contours sont moins bien définis à cause de cette définition vectorielle. Cependant, la qualité des contours détectés peut être améliorée, par exemple dans le cas de la couleur, par l'utilisation des trois bandes plutôt que seulement de l'intensité [10]. En effet, un processus de détection de contours couleurs devrait reconnaître les changements dans l'intensité lumineuse, dans la saturation et dans la teinte [29]. Cependant, les aspects saturation et teinte sont perdus lorsque l'image couleur est convertie en niveaux de gris et donc certains contours, distinguant deux régions de même intensité mais avec des teintes ou des saturations différentes, sont perdus (figure 1.4). Ce problème n'est pas spécifique à la couleur. Le satellite Landsat-TM fournit sept bandes, acquises à des longueurs d'onde différentes et représentant la même scène. Le problème qui se pose alors concerne la fusion des informations pertinentes de chaque bande. Au moins deux solutions sont possibles. La première consiste à extraire l'information dans chacune des bandes séparément (par exemple détecter les contours de chaque bande) et ensuite à fusionner le résultat de ces traitements. Cette solution semble complexe à mettre en oeuvre [7, 10]. La seconde consiste à fusionner les bandes pendant le traitement. Par exemple, étendre la notion du contour, bien connue dans le cas des images à niveaux de gris, aux images multispectrales. Nous avons opté pour cette

solution.

Nous avons mentionné précédemment que dans une image multispectrale, chaque point est représenté par un vecteur. Nous tentons de définir un contour multispectral. Prenons, par exemple, le cas d'une image discrète 1D à deux bandes ($\mathbb{R} \Rightarrow \mathbb{R}^2$). À chaque pixel est associé un vecteur de composantes $(I_1(x), I_2(x))$. Un contour est engendré par une variation brusque de type marche dans les modules des pixels et/ou les directions des pixels. Pour détecter ces contours, une solution possible est de mesurer la longueur du vecteur de la différence entre deux pixels voisins (figure 1.2). Dans le cas continu, nous pouvons remplacer le vecteur de différence par le vecteur de dérivée. Ce concept est généralisable au cas à m bandes. Donc le contour multispectral peut être détecté par un maximum de la longueur du vecteur de dérivée des points de l'image. Dans le cas 2D, nous pouvons calculer le vecteur de la dérivée directionnelle des points et chercher la direction où le module de ce vecteur est maximal. Suivant cette définition de contour, il existe un détecteur gradient [13, 15] ainsi qu'une deuxième dérivée directionnelle [8], mais aucun Laplacien n'a encore été défini. Mentionnons que le Laplacien, est aussi utile dans la détection de certains types de jonctions ainsi que dans la reconstruction 3D [11, 32].

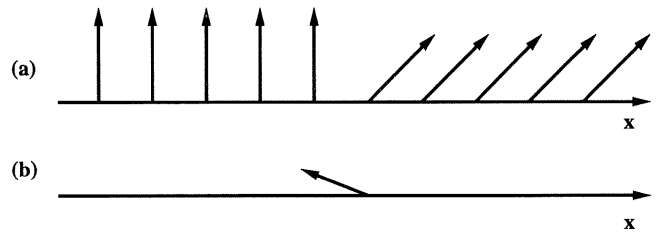


Figure 1.2 – *(a) Contour de type marche dans une image 1D à 2 bandes. (b) Vecteurs de différence entre deux pixels voisins.*

Nous présentons dans ce chapitre le développement d'un opérateur Laplacien multispectral. Dans la section 1.2, nous présentons les travaux existants dans le domaine de la détection de contours multispectraux. Dans la section 1.3, nous développons le Laplacien multispectral à partir d'une fonction de contraste présentée par Cumani [8, 9]. Dans

la section 1.4, nous discutons de quelques résultats expérimentaux. Finalement, nous concluons en section 1.5.

1.2 Travaux existants

Beaucoup d'efforts ont été mis sur la détection de contours dans les images à niveaux de gris, mais les images multispectrales ont été un peu négligées. Il existe principalement deux types d'approche. La première, la plus couramment utilisée, consiste à trouver les contours dans les différentes images et ensuite à effectuer une fusion des images de contours résultantes. Dans le contexte des images couleurs (trois bandes), Delcroix et Abidi [10] effectuent d'abord une détection de contours sur chaque bande de façon indépendante, ensuite, ils effectuent une fusion des trois images de contours par une fonctionnelle, avec des contraintes sur la normalisation, la plausibilité et la corrélation des données. Chu et Aggarwal [7] estiment par maximum de vraisemblance une solution initiale pour la position des contours fusionnés puis raffinent cette solution par une minimisation de la courbure des contours. Ensuite les régions résultantes, entourées par ces contours, sont fusionnées pour créer des régions compactes et suffisamment grandes.

La fusion des contours donne lieu à des procédés complexes, c'est pourquoi la seconde approche consiste plutôt à trouver une seule image de contours directement de l'image multispectrale. Djurié et Fwu [14] utilisent la théorie Bayesienne pour sélectionner la meilleure hypothèse quant au nombre de contours et à leur localisation, et ce, directement à partir de l'image multispectrale. Scharcanski et Venetsanopoulos [29] développent un gradient couleur défini comme une différence dans les statistiques locales du champ vectoriel représentant l'image. Di Zenzo [13], Novak et Shafer [28], Lee et Cok [25], Cumani [8, 9] et Drewniok [16, 15] développent un autre gradient multispectral par différentes méthodes. Ce gradient permet aux différentes bandes de coopérer entre elles, c'est-à-dire

qu'un contour dans une direction, dans une bande, renforce (une plus grande plausibilité) un contour occupant la même position dans la même direction, mais dans une autre bande. Di Zenzo utilise les tenseurs pour définir ce gradient. Novak et Shafer utilisent la matrice du Jacobien mais uniquement pour trois bandes. Lee et Cok utilisent les champs vectoriels. De plus, ils prouvent que lorsque les signaux des différentes bandes sont correllés, le gradient multispectral donne un meilleur rapport signal/bruit que la somme des carrés des gradients individuels. Cumani obtient le même gradient par calcul différentiel. De plus, il définit une mesure de contraste directionnel au carré que nous utilisons pour dériver notre Laplacien multispectral. En utilisant cette mesure de contraste, il définit la deuxième dérivée dans la direction du gradient. Les contours correspondent aux passages par zéro de cette dérivée. Il obtient cependant une indétermination de signe qui génère plusieurs faux passages par zéro qui ne sont pas des contours. Il tente d'éliminer ces passages par zéro par une méthode subpixel. Finalement, Drewniok généralise au multispectral le gradient couleur de Novak et Shafer, par la matrice du Jacobien.

Pour définir le Laplacien multispectral, nous dérivons la fonction de contraste de Cumani dans une direction quelconque $\vec{u}$ afin d'obtenir une deuxième dérivée directionnelle. Celle-ci permet de définir le Laplacien multispectral.

1.3 Laplacien multispectral

Une image multispectrale est formée de plusieurs bandes (e.g. trois images dans le cas de la couleur, sept dans le cas Landsat-TM). Elle peut être vue comme la fonction $\mathbf{f} : \mathbb{R}^2 \Rightarrow \mathbb{R}^m$ où m est le nombre de bandes. Cette fonction fait correspondre à un point (x, y), un vecteur de fonctions $\mathbf{f}(x, y) = (f_1(x, y), f_2(x, y), \ldots, f_m(x, y))$. Ainsi, pour $m = 1$, nous avons le cas des images monochromatiques. Pour plus de lisibilité, nous omettrons les (x, y) dans les prochaines expressions.

1.3.1 Fonction de contraste

Dans cette section, nous rappelons les définitions nécessaires pour l'élaboration de notre Laplacien multispectral. L'idée principale consiste à chercher une mesure du contraste dans l'image et l'utiliser pour définir un Laplacien. Le contraste multispectral est défini comme étant une mesure de la longueur du vecteur des dérivées et ce, dans une direction $\vec{u}$ quelconque. Cette longueur peut être calculée par une norme (module). Ce vecteur a m composantes et correspond à la première dérivée directionnelle de $\mathbf{f}$ dans la direction $\vec{u} = (u_1, u_2)$:

$$\frac{\partial \mathbf{f}}{\partial \vec{u}} = \begin{pmatrix} \frac{\partial f_1}{\partial \vec{u}} \\ \frac{\partial f_2}{\partial \vec{u}} \\ \vdots \\ \frac{\partial f_m}{\partial \vec{u}} \end{pmatrix}.$$

Nous cherchons la direction $\vec{\eta}$, qui correspond à la plus grande pente de la fonction, c'est-à-dire telle que $\|\frac{\partial \mathbf{f}}{\partial \vec{\eta}}\|$ est maximale. $\|\vec{v}\|$ représente la norme ou le module de $\vec{v}$. Dans le cas de la norme Euclidienne, nous avons:

$$D_{\vec{u}} = \|\frac{\partial \mathbf{f}}{\partial \vec{u}}\|^2 = Eu_1^2 + 2Fu_1u_2 + Gu_2^2 \tag{1.1}$$

où

$$E = \sum_{i=1}^{m} \left(\frac{\partial f_i}{\partial x}\right)^2, \qquad F = \sum_{i=1}^{m} \frac{\partial f_i}{\partial x}\frac{\partial f_i}{\partial y}, \qquad G = \sum_{i=1}^{m} \left(\frac{\partial f_i}{\partial y}\right)^2.$$

$D_{\vec{u}}$ définit donc une mesure du contraste au carré dans une direction $\vec{u}$ quelconque.

Le contraste maximal, λ_{max}, correspond à la plus grande valeur propre de $\mathbf{J}^t\mathbf{J}$, où $\mathbf{J}$ est la matrice Jacobien de $\mathbf{f}$ et est définie par

$$\mathbf{J} = \begin{pmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \\ \vdots & \vdots \\ \frac{\partial f_m}{\partial x} & \frac{\partial f_m}{\partial y} \end{pmatrix}$$

et $\mathbf{J}^t$ est sa transposée. La direction de cette variation maximale, $\vec{\eta}$ correspond au vecteur propre associé à λ_{max}. Nous avons:

$$\mathbf{J}^t\mathbf{J} = \begin{bmatrix} E & F \\ F & G \end{bmatrix}$$

et donc,

$$\lambda_{max} = \frac{E + G + \sqrt{(E-G)^2 + 4F^2}}{2} \quad \text{et} \quad \vec{\eta} = \pm(F, \lambda_{max} - E). \tag{1.2}$$

Donc, λ_{max} et $\vec{\eta}$ sont le module au carré et la direction du gradient multispectral. Notons que la composante en y de $\vec{\eta}$, soit $\lambda_{max} - E$, est toujours positive alors que la composante en x, soit F, peut être positive ou négative. Le lecteur intéressé pourra trouver un complément dans [8, 13, 15].

1.3.2 Définition du Laplacien multispectral

Dans cette section, nous développons notre Laplacien multispectral. Le Laplacien est un opérateur du second ordre qui peut être défini comme $f_{\vec{v}\vec{v}} + f_{\vec{p}\vec{p}}$ où $f_{\vec{v}\vec{v}}$ et $f_{\vec{p}\vec{p}}$ sont les deuxièmes dérivées directionnelles de $\mathbf{f}$ dans une direction $\vec{v} = (v_1, v_2)$ quelconque, et dans la direction $\vec{p} = (-v_2, v_1)$ perpendiculaire à $\vec{v}$ respectivement. La deuxième dérivée directionnelle de $\mathbf{f}$ dans une direction $\vec{u}$ quelconque peut être définie, à partir de la fonction de contraste multispectral au carré (équation 1.1), de la façon suivante:

$$\mathbf{f}_{\vec{u}\vec{u}} = \frac{\partial(\pm\sqrt{D_{\vec{u}}})}{\partial\vec{u}} = \pm\left(\frac{E_x u_1^3 + (2F_x + E_y)u_1^2 u_2 + (2F_y + G_x)u_1 u_2^2 + G_y u_2^3}{2\sqrt{D_{\vec{u}}}}\right) \tag{1.3}$$

où E_x, E_y, F_x, F_y, G_x et G_y représentent les dérivées en x et en y de E, F et G respectivement. Étant donné que nous dérivons la valeur absolue de la fonction de contraste, il reste donc une indécision quand au signe de la dérivée seconde. L'équation du Laplacien

est donc:

$$L = \mathbf{f}_{\vec{v}\vec{v}} + \mathbf{f}_{\vec{p}\vec{p}} = \pm\left(\frac{E_x v_1^3 + (2F_x + E_y)v_1^2 v_2 + (2F_y + G_x)v_1 v_2^2 + G_y v_2^3}{2\sqrt{D_{\vec{v}}}}\right) \pm\left(\frac{-E_x v_2^3 + (2F_x + E_y)v_2^2 v_1 - (2F_y + G_x)v_2 v_1^2 + G_y v_1^3}{2\sqrt{D_{\vec{p}}}}\right). \quad (1.4)$$

À cause de la géométrie du plan image, les meilleures directions à utiliser sont les directions des axes x et y. Pour ce faire, nous remplaçons $\vec{v} = (1, 0)$, soit l'axe des x, et $\vec{u} = (0, 1)$, soit l'axe des y, dans l'équation 1.4. Nous trouvons donc une expression simplifiée pour le Laplacien correspondant à $\mathbf{f}_{xx} + \mathbf{f}_{yy}$:

$$L = \frac{1}{2}\left(\left(\pm\frac{E_x}{\sqrt{E}}\right) + \left(\pm\frac{G_y}{\sqrt{G}}\right)\right) \quad (1.5)$$

Nous avons quatre possibilités pour L. Nous devons choisir une de ces possibilités en déterminant le signe des contrastes en x et y, car c'est le signe du contraste qui détermine le signe de l'équation 1.3. Selon l'équation 1.1, les contrastes en x et y sont $\pm\sqrt{E}$ et $\pm\sqrt{G}$ respectivement. La composante en x de l'orientation du gradient, $\vec{\eta}$, nous indique le signe du contraste en x et la composante en y de $\vec{\eta}$ nous indique le signe du contraste en y. Étant donné que $\lambda_{max} - E$ est toujours positif, nous fixons le contraste en y à $+\sqrt{G}$. Le signe de F (voir équation 1.2) nous permet donc de déterminer le signe du contraste en x par rapport à celui en y, soit $\pm\sqrt{E}$. Maintenant, nous pouvons exprimer L de la façon suivante:

$$L = \begin{cases} \pm\frac{1}{2}\left(\frac{E_x}{\sqrt{E}} + \frac{G_y}{\sqrt{G}}\right) & \text{si } F \geq 0 \\ \pm\frac{1}{2}\left(-\frac{E_x}{\sqrt{E}} + \frac{G_y}{\sqrt{G}}\right) & \text{si } F < 0. \end{cases} \quad (1.6)$$

1.3.3 Problème de l'orientation

Selon l'équation 1.6, nous avons encore deux possibilités pour le Laplacien, soit $\pm|L|$. Ceci est dû au fait que l'orientation du gradient multispectral, $\vec{\eta}$, est définie comme un

vecteur propre et donc, son signe n'est pas spécifié (voir équation 1.2). Étant donné que la composante en y de $\vec{\eta}$, $\lambda_{max}-E$, est positive, l'orientation calculée se situe dans l'intervalle $[0, \pi]$. Il y a donc une confusion entre θ et $\theta+\pi$. Selon la procédure décrite précédemment pour obtenir l'équation 1.6, ce sont les signes des composantes de $\vec{\eta}$ qui déterminent le signe du Laplacien. La confusion d'orientation de $\vec{\eta}$ cause donc une inversion du signe du Laplacien. Prennons par exemple deux pixels voisins dont les Laplaciens sont de signes opposés. Nous avons deux cas possibles. Le premier cas est qu'un contour passe entre les deux pixels et alors les orientations des gradients sont cohérentes (i.e. la différence entre les deux orientations est inférieure à un seuil). Le deuxième cas est celui où aucun contour ne passe entre les deux pixels. La différence de signes des Laplaciens est donc dûe à une incohérence dans les orientations. Dans le deuxième cas, il est possible de corriger le Laplacien en inversant son signe pour un des deux pixels. Ces situations de changements brusques d'orientation se font essentiellement aux endroits où la tangeante du contour est verticale car l'intervalle de l'orientation est $[0, \pi]$ (voir figures 1.3, 1.5.a et 1.5.b).

La procédure que nous utilisons pour corriger le Laplacien se base sur le fait que dans les endroits où le gradient a un sens (régions voisines des contours de type marche), il ne devrait pas y avoir de changements brusques de son orientation, c'est-à-dire supérieur à un seuil. Nous fixons ce seuil pour les changements brusques d'orientation à $\pi/2$. Si dans ces endroits, nous trouvons un tel changement, nous modifions l'orientation de π. Cette correction ne concerne pas les régions d'intensité constante puisque le vecteur gradient est nul. Ainsi, nous n'effectuons la correction que dans les régions autour des contours, c'est-à-dire où λ_{max} est supérieur à un certain seuil. Nous prennons un pixel appartenant à une de ces régions et, considérant que son orientation est correcte, nous propageons la correction dans toute la

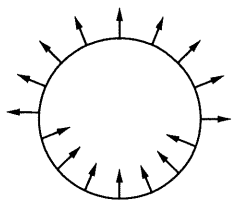


Figure 1.3 – *Exemple de contour avec l'orientation du gradient multispectral à quelques points.*

région grâce à un algorithme de remplissage de taches utilisé en infographie [19].

1.3.4 Algorithme

L'algorithme de détection de contours sur une image multispectrale par le Laplacien comporte quatre étapes. La première consiste à calculer les dérivées premières et secondes de chaques bandes en x et y à l'aide de convolutions de ces bandes avec les dérivées partielles de la Gaussienne de paramètre σ. Ensuite, pour chaque pixel de l'image, nous calculons le module et l'orientation du gradient, λ_{max} et $\vec{\eta}$, à l'aide de l'équation 1.2, et le Laplacien, L, à l'aide de l'équation 1.6. La troisième étape consiste à corriger le Laplacien. Pour ce faire, nous balayons l'image pour trouver un pixel non traité où λ_{max} est supérieur à un seuil S. Ce pixel sert de germe pour la correction. Nous marquons ce pixel comme étant traité. Nous balayons colonne par colonne (ou ligne par ligne) la région définie par tous les pixels connexes au germe et dont λ_{max} est supérieur à S. Ce balayage s'effectue en partant du germe et en s'assurant que chaque paire de pixels comparés contient un pixel traité et un non traité. Pour chaque paire de pixels, si la différence d'orientation est supérieure à $\pi/2$ et inférieure à $3\pi/2$, alors nous modifions l'orientation de π et inversons le signe de L du pixel non traité. Ensuite, nous marquons ce pixel comme étant traité. Nous continuons à balayer l'image et appliquons la correction à une autre région jusqu'à ce que le balayage de l'image soit complété. La quatrième étape consiste à détecter les passages par zéro du Laplacien. Pour chacun des pixels, s'il n'est pas déjà marqué comme un passage par zéro, nous comparons dans quatre directions (0, $\pi/4$, $\pi/2$ et $3\pi/4$). Quand il y a changement de signe dans une des directions, nous marquons le pixel, dont la valeur absolue du Laplacien est la moins élevée, comme étant un passage par zéro. Le résultat final de cet algorithme est une image binaire indiquant la position des contours détectés.

Dans cet algorithme, il y a deux paramètres à fixer. Le premier est le paramètre σ de

la Gaussienne. Ce paramètre influence le voisinage sur lequel les dérivées sont calculées. c'est-à-dire le niveau de lissage de l'image. Le deuxième paramètre est le seuil S sur λ_{max}. Ce seuil détermine sur quels pixels s'effectuera la correction du Laplacien. Plus ce seuil est bas et plus il y a de pixels dans la correction. À ces deux paramètres propres à l'algorithme, peut être ajouté un deuxième seuillage sur λ_{max} servant à éliminer les points de contours correspondants au bruit.

1.4 Résultats

Dans cette section, nous discutons des résultats obtenus sur des images synthétiques et réelles avec notre Laplacien multispectral.

La figure 1.4 montre une image couleur synthétique bruitée (figures 1.4a, 1.4b, 1.4c) fabriquée de façon à ce que l'intensité soit constante (figure 1.4d). Il est donc évident que l'utilisation d'un détecteur Laplacien sur l'image d'intensité ne fournirait aucun contour. L'image 1.5a montre le Laplacien multispectral avant correction. Nous pouvons noter le problème d'orientation décrit à la section 1.3.3. À cause du bruit, le Laplacien oscille entre le positif et le négatif dans les régions de contours verticaux. La figure 1.5b montre les passages par zéro de l'image 1.5a. Nous remarquons que nous avons des faux passages par zéro qui ne sont pas dûs au bruit. La figure 1.5c montre les passages par zéro après la correction du Laplacien. Cette correction induit de nouveaux contours qui sont en fait les frontières des régions de correction. Ces contours possèdent un λ_{max} faible et donc ils peuvent être éliminés par un seuillage (figure 1.5d). La figure 1.7a présente le résultat du Laplacien multispectral sur une image couleur (figures 1.6a, 1.6b et 1.6c). La figure 1.7b présente les contours obtenus à partir du Laplacien de l'image d'intensité de cette même image (figure 1.6d). Il est à noter que les deux images de contours sont semblables. Ceci montre la validité de notre Laplacien multispectral dans le cas des contours de type

marche. Par contre, les lignes (voir côtés de 1.7a) ont généré deux contours indésirables. Ceci est dû au fait qu'aux points centraux des lignes, la dérivée première est nulle et cela fausse la correction du signe du Laplacien. Comme nous le voyons, cette nouvelle technique donne d'excellents résultats sur cette image simple. Malheureusement, ce n'est pas le cas sur des images complexes. En effet, la correction du Laplacien est sensible au seuil (figures 1.9a et 1.9b) c'est-à-dire qu'elle a tendance à induire de nouveaux contours aux frontières des zones de correction. Ces faux contours sont difficiles à éliminer par un seuillage simple dans le cas des images de texture car parfois ils ont un λ_{max} plus élevé que celui des vrais contours. Cependant, nous pouvons remarquer que les triangles du chapeau sont mieux détectés en 1.9a et 1.9b qu'en 1.9c.

1.5 Conclusion

Dans ce chapitre, nous avons présenté un nouveau détecteur de contours Laplacien pour les images multispectrales construit à partir de la fonction de variation de constraste présentée par Cumani [8]. Le développement d'un tel Laplacien peut être utile dans plusieurs thèmes de la vision par ordinateur et du traitement d'image: détection de jonctions, reconstruction 3D, etc.

Ce Laplacien multispectral donne des résultats valides à un signe près. Cependant il est sensible au seuil et présente un problème dans le cas des contours de type ligne. Pour réduire la sensibilité au seuil, la détermination des régions de correction pourrait être améliorée. Dans l'optique de cerner le problème des lignes et autre types de contour, il serait intéressant de vérifier le comportement du Laplacien multispectral en considérant différents types de contours.

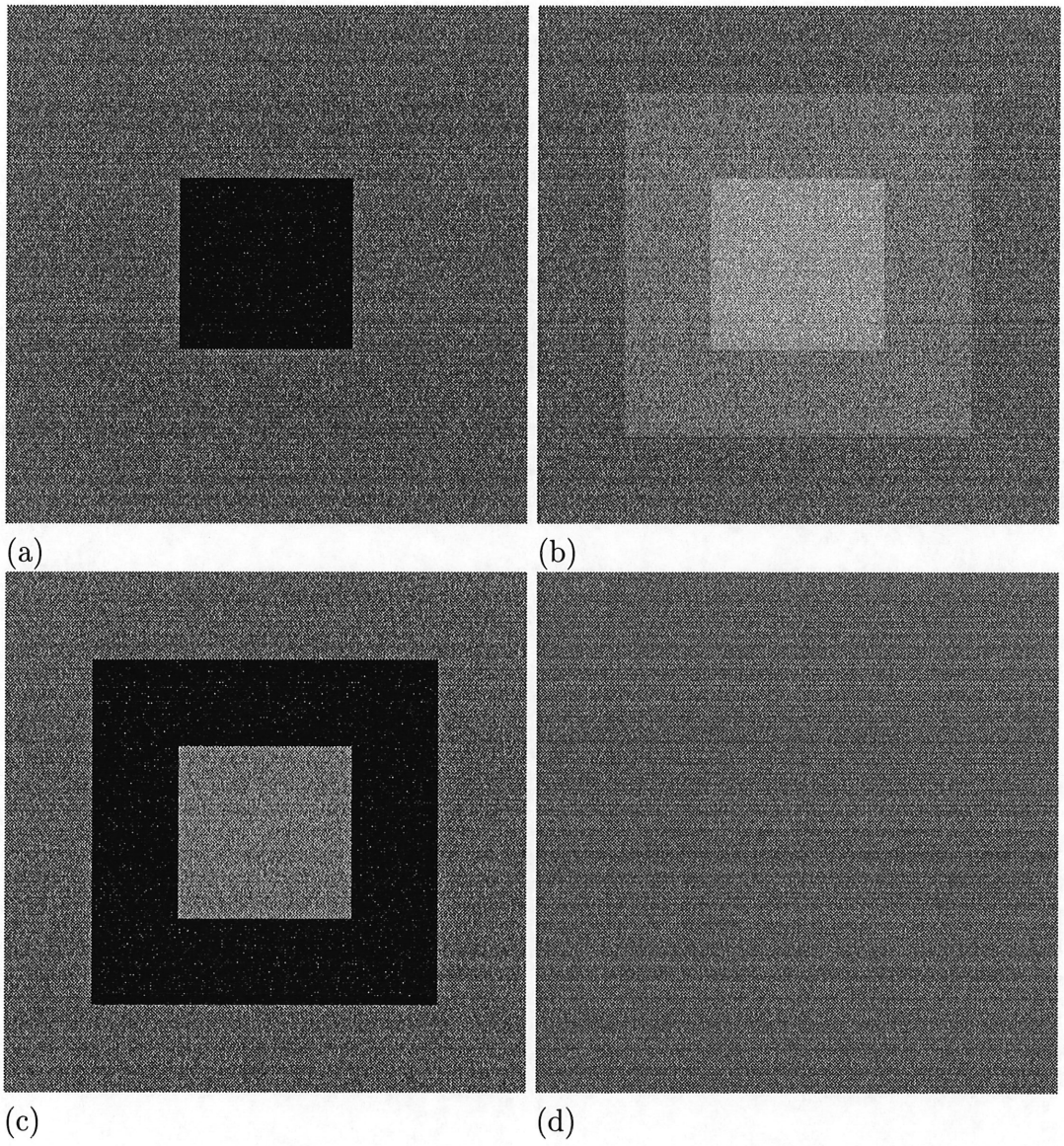


Figure 1.4 – *Image synthétique. (a) Bande bleue. (b) Bande verte. (c) Bande rouge. (d) Image d'intensité.*

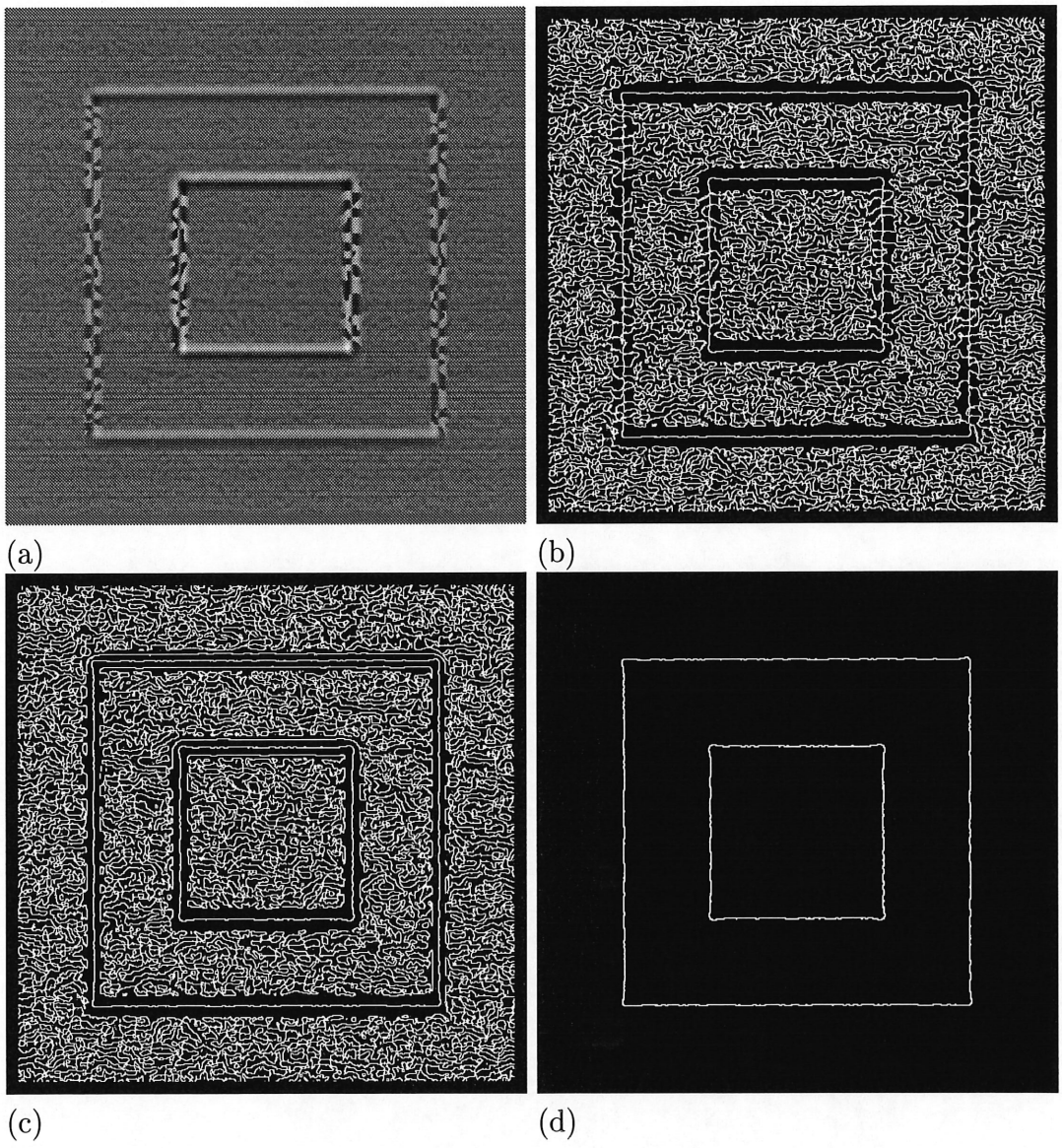


Figure 1.5 – *Résultats sur l'image couleur 1.4 (1.4a, 1.4b, 1.4c). (a) Laplacien multispectral non corrigé ($\sigma = 4$). (b) Passages par zéro de a. (c) Passages par zéro du Laplacien corrigé. Les pixels corrigés sont ceux dont le module de gradient, λ_{max}, est dans les 10% supérieur. (d) Passages par zéro après un seuillage à 92% du module du gradient.*

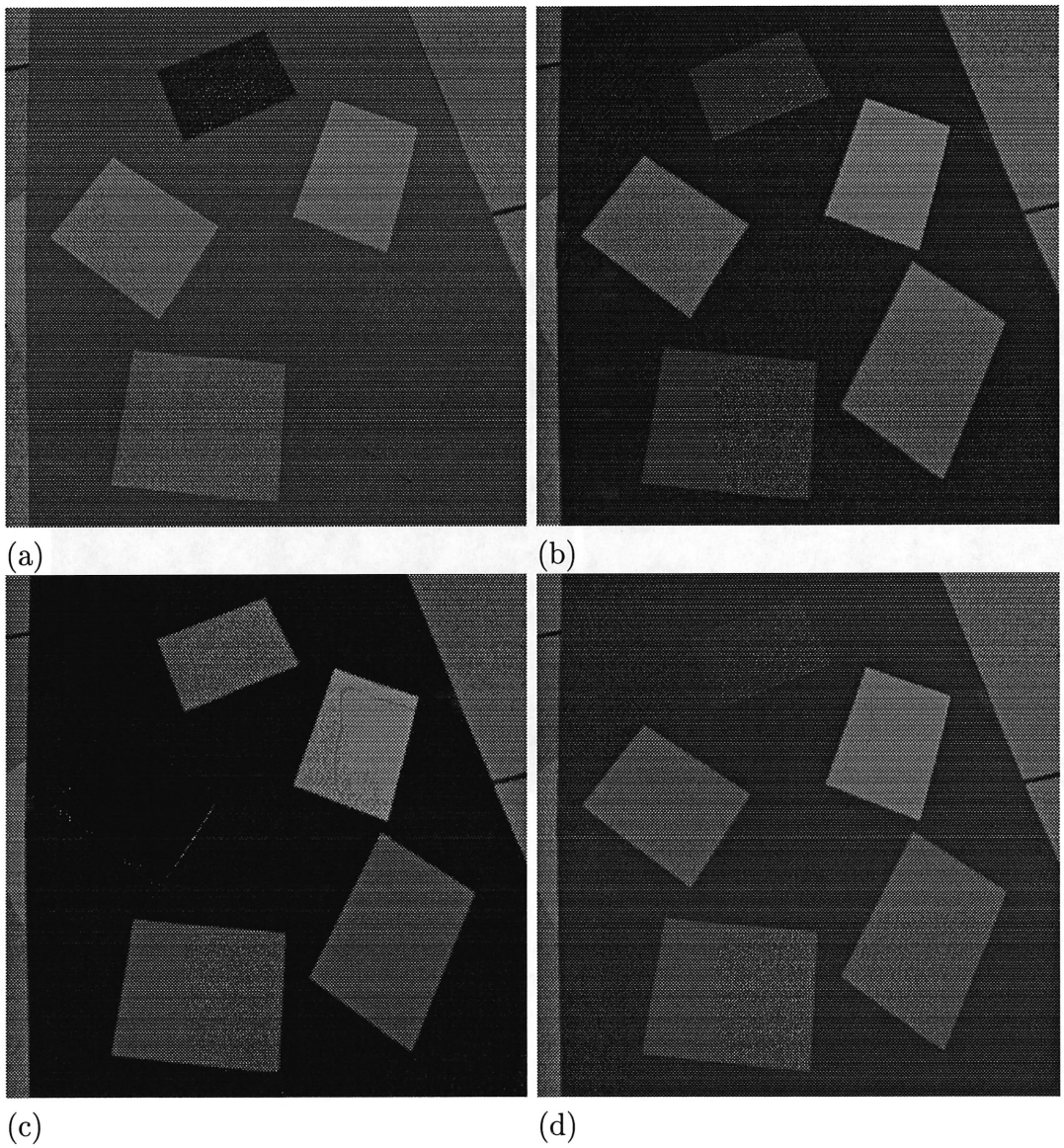
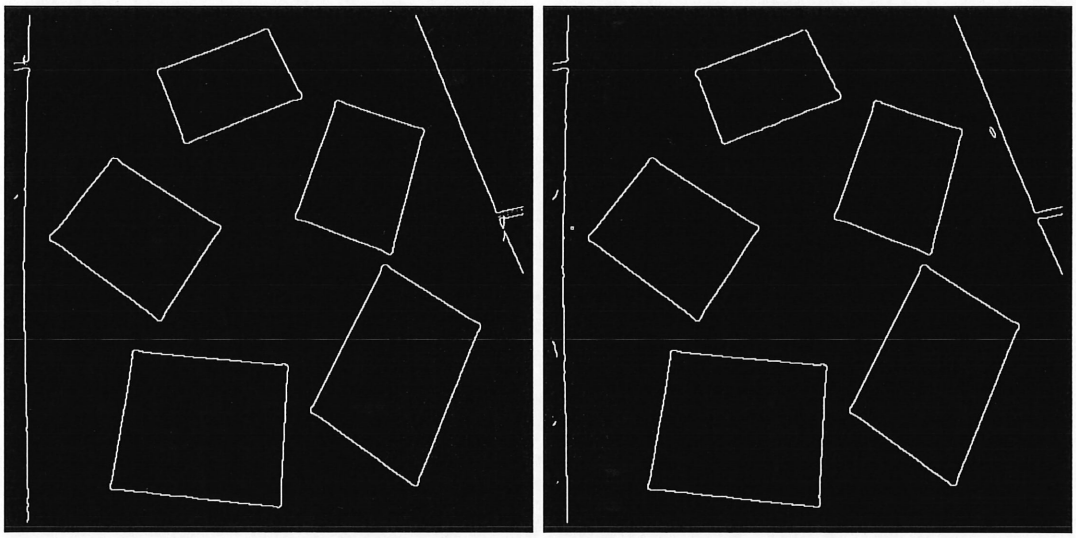


Figure 1.6 – *Image réelle de papier (INRIA-Syntim ©copyright). (a) Bande rouge. (b) Bande verte. (c) Bande bleue. (d) Image d'intensité.*



(a) (b)

Figure 1.7 – *(a) Contours de l'image couleur obtenus par le Laplacien multispectral ($\sigma = 3$) sur l'image de papier (1.6a, 1.6b et 1.6c). (b) Contours de l'image d'intensité obtenus par le Laplacien sur l'image 1.6.d ($\sigma = 3$).*

Figure 1.8 – *Image réelle d'Ariane. (a) Bande rouge. (b) Bande verte. (c) Bande bleue. (d) Image d'intensité.*

(a) (b)

(c)

Figure 1.9 – *(a) Passages par zéro du Laplacien multispectral non corrigé de l'image d'Ariane (1.8.a, 1.8.b et 1.8.c). (b) Passages par zéro du Laplacien multispectral corrigé. (c) Passages par zéro du Laplacien de l'image 1.8.d.*

CHAPITRE 2

DÉTECTION DE CONTOURS DE TEXTURE

2.1 Introduction

La texture est un élément omniprésent dans la réalité. Chaque surface possède sa texture propre. La texture est donc un indice important dans la perception de la nature de la surface observée. L'être humain utilise la texture pour la perception de l'environnement, en particulier la reconstruction 3D. Nous distinguons aussi les frontières entre les régions de textures différentes, même si l'intensité moyenne est la même dans les deux régions [18]. Dans certains domaines d'applications de la vision artificielle, il est essentiel de pouvoir détecter les frontières entre les différentes régions de texture. Pensons simplement à l'analyse de scènes où la texture est utilisée pour identifier les différents types d'objets. En imagerie médicale, la texture peut servir à caractériser les tissus et à l'aide au diagnostique [34]. En recherche d'images par le contenu, la texture est un élément important pour trouver les images semblables à une image donnée [1]. Mentionnons, comme autres

exemples d'applications, l'inspection automatique et l'analyse de documents [18, 34]. Malheureusement, dans plusieurs méthodes utilisées pour la détection de contours, l'aspect texture est souvent négligé. Des hypothèses simplificatrices sur les niveaux de gris sont prises en compte et, le plus souvent, c'est le niveau de gris moyen seulement qui est analysé [22, 34]. Par conséquent, ces méthodes, appliquées aux images de texture, résultent souvent en une abondance de contours (entre les éléments de texture) tout en ne détectant pas clairement les frontières entre les régions (figures 2.7b et 2.8b).

Ce chapitre présente une nouvelle approche pour la détection de contours entre les régions texturées. D'abord, nous présentons un survol de l'analyse de texture en détaillant plus particulièrement la détection de contours. Nous présentons ensuite notre nouvelle approche basée sur les moments locaux. Ensuite, nous présentons quelques résultats avant de conclure.

2.2 L'analyse de texture

La texture étant un élément important de la perception humaine, son analyse joue un rôle essentiel dans certaines applications. Dans cette section, nous présentons quelques définitions de la texture, un survol des méthodes de représentation de texture, les problèmes reliés à l'analyse de texture et parmi ces problèmes, nous détaillons plus particulièrement la détection de contours entre régions de texture.

2.2.1 Définition de la texture

Par les exemples présentés en introduction, nous voyons que la texture peut servir de différentes manières. Cette réalité se reflète sur la définition de texture. En effet, aucun concensus ne s'est formé car chacun utilise la définition de texture qui sied le mieux à son

application. Il n'existe donc aucune définition universelle de la texture [35]. Tuceryan et Jain [34] citent quelques définitions prises dans différents articles qui démontrent bien ce fait. Une de ces définitions présente la texture comme une structure d'éléments répétitifs, appelés primitives, arrangés selon une règle de placement déterministe ou stochastique. La texture implique une distribution spatiale des niveaux de gris. Elle peut être perçue à diverses résolutions de l'image, l'exemple classique de ce phénomène étant le mur de briques. À une distance lointaine, nous apercevons un ensemble de briques (macrotextures) placées de façon déterministe, mais à une distance plus rapprochée (quelques briques dans le champ de vision), le grain de la brique est maintenant visible et est perçu comme une texture (microtexture) aléatoire. Une région est considérée texturée si le nombre de primitives présentes dans cette région est grand. La texture peut être décrite par des attributs statistiques, syntaxiques, morphologiques ou qualificatifs comme uniformité, densité, grossièreté, linéarité et directionnalité.

2.2.2 Méthodes de représentation de texture

Comme la définition de texture varie d'une application à l'autre, il est évident qu'aucune méthode de représentation de texture n'est adéquate pour toutes les applications [34]. Il existe trois grandes catégories de méthodes: statistiques, basées sur un modèle, et de traitement de signal

- Les méthodes statistiques ont été utilisées très tôt étant donnée la nature aléatoire de la texture. Dans ces méthodes, la texture est caractérisée par les statistiques de premier ordre, calculées à partir des niveaux de gris ou de leur distribution, telles la moyenne, la variance, la médiane ou les moments d'ordre supérieur.

  Les méthodes statistiques comprennent aussi des statistiques du second ordre sur la matrice de co-occurrence. Cette matrice fait ressortir certaines propriétés sur la

distribution spatiale des niveaux de gris. Le nombre de paires de pixels situés à une distance d, selon une direction θ, ayant des niveaux de gris i et j, est l'entrée i, j de la matrice. Donc, il faut calculer une matrice pour chaque d et chaque θ voulus. Plusieurs statistiques sont calculées à partir de cette matrice telles l'énergie, l'entropie, l'inertie, l'homogénéité locale et la corrélation [35]. Des caractéristiques perceptuelles comme le contraste, l'homogénéité, la grossièreté et la directionnalité sont aussi calculées à partir de la matrice de co-occurrence [2].

- Les méthodes basées sur un modèle permettent, non seulement de décrire la texture, mais aussi de la synthétiser. Parmis ces méthodes, il y a le modèle autorégressif. Ce modèle stochastique permet de modéliser les interactions spatiales entre les pixels d'une image [1]. L'intensité du pixel dépend de ses voisins et les erreurs commises à chaque pixel sont supposées indépendantes. Un deuxième modèle, le modèle des champs markoviens, suppose aussi que l'intensité à un pixel dépend uniquement de ses voisins. Par contre, les erreurs commises sont supposées dépendantes. Ces modèles représentent bien les microtextures mais moins bien les macrotextures, c'est pourquoi il est préférable d'effectuer un test d'homogénéité avant de les utiliser [34]. Ces modèles sont estimés en évaluant des paramètres de texture. En dernier lieu, le modèle fractal tente de représenter les textures qui présentent la particularité d'avoir les mêmes caractéristiques, peu importe la résolution [34]. Un exemple classique de fractale naturelle est la fougère.

- Plusieurs études psychophysiques montrent que le système visuel humain effectue une analyse spatiale et fréquentielle par filtrage de l'image [27, 34]. Suivant cette approche, certaines propriétés des images filtrées sont calculées telles: le spectre de Fourier, la phase et les points stationnaires de la phase. Pour conserver les propriétés locales de l'image, certains auteurs utilisent le filtre de Gabor [34].

2.2.3 Problèmes reliés à l'analyse de textures

Il existe plusieurs thèmes de la vision artificielle et du traitement d'images dans lesquels il s'effectue de l'analyse de texture. Citons la classification, la segmentation et la reconstruction 3D. Nous présentons ces trois thèmes brièvement dans cette section.

- La classification de texture consiste à décider à quelle catégorie de texture l'image appartient [34]. Il faut donc avoir des connaissances *a priori* des catégories de texture. En premier lieu, les paramètres du modèle de texture sont déterminés et ensuite, les techniques de classification sont appliquées. Une partie importante du travail de classification consiste donc en fait à trouver un modèle adéquat pour représenter la texture. La classification est la méthode la plus utilisée dans les applications comportant de la texture. Par exemple, elle est utilisée en analyse de scènes pour tenter de discriminer les différents objets.
- La segmentation de texture consiste à séparer les différentes régions texturées dans une image. Cette étape peut être utilisée avant d'effectuer une classification de texture. Deux approches sont utilisées, similairement à la segmentation régulière, c'est-à-dire l'approche basée sur les régions, et l'approche basée sur les contours [34]. La première approche consiste à regrouper les pixels en petites régions qui ont les mêmes propriétés de texture. Ces régions sont ensuite fusionnées itérativement jusqu'à l'obtention de plus grandes régions homogènes par rapport à un critère prédéfini. Les différentes régions sont séparées par des contours fermés. Cependant la segmentation en région est sensible à la manière de regrouper les pixels (partition initiale). Elle souffre du problème de sous-segmentation ou de sur-segmentation. Les différentes régions sont toujours séparées et leurs contours fermés. L'approche basée sur les contours consiste à localiser les frontières des régions de texture adjacentes. Cette approche produit des frontières de régions mieux localisées, cependant les

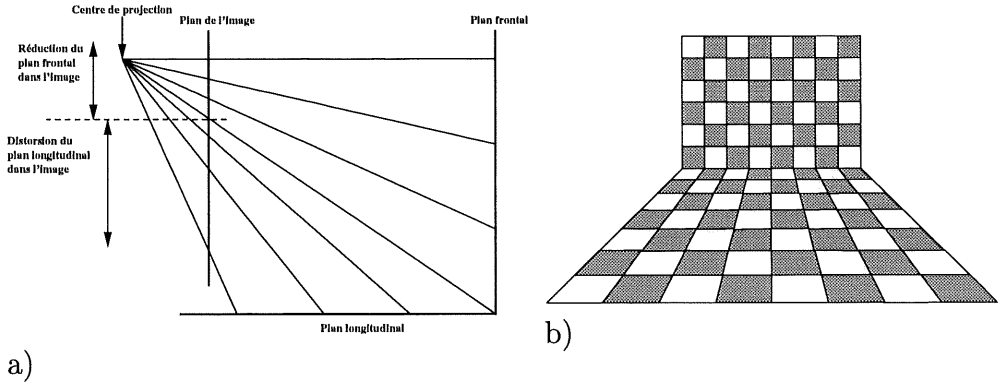


Figure 2.1 – *Réduction et compression*

contours ne sont pas automatiquement fermés. Cette approche peut être duale à la fusion si elle produit des contours fermés. Dans la section 2.2.4, nous allons détailler quelques approches existantes pour détecter les contours de texture, puisque la détection de contours de texture constitue la finalité de ce chapitre.

- Les variations de texture sont des indices de la forme des objets. Effectivement, les primitives de texture se déforment sous l'effet de la projection perspective (voir figure 2.1). En mesurant cette déformation, il est possible de retrouver l'orientation des surfaces.

2.2.4 Contours de texture

Les contours de texture peuvent être définis comme la frontière entre deux régions ayant des textures perceptuellement différentes [33]. Habituellement, les détecteurs Laplacien et gradient sont utilisés pour extraire les contours de type marche; c'est-à-dire les contours correspondant à un point d'inflexion de l'image, convoluée avec un filtre passe-bas. Cependant, il existe des contours visibles ne correspondant pas à un point d'inflexion de l'image lissée (figure 2.7a). En d'autres termes, la moyenne locale des niveaux de gris

d'une telle image reste constante [33]. Un détecteur Laplacien ou gradient ne trouve donc pas ces contours.

Pour cela, des détecteurs de contours de texture ont été développés [33, 30, 22, 18, 27, 21, 23, 17, 36]. Parmi les détecteurs de contours de texture existants, mentionnons les suivants. Thompson [33] a développé un détecteur de contours incorporant plusieurs indices de texture. Ce système est basé sur un opérateur de différences entre deux régions. Les différences de plusieurs caractéristiques de texture sont combinées en somme pondérée. Ensuite, les contours sont trouvés aux maxima de ces différences. Souza [30] calcule des tests statistiques basés sur un voisinage pour chaque point d'un signal unidimentionnel. Certains de ces tests permettent la localisation des frontières de régions ayant des moyennes différentes, d'autres, les frontières de régions ayant des moyennes égales mais des variances différentes. Kashyap et Eom [22] utilisent un modèle de corrélation pour estimer les paramètres de texture dans un petit voisinage. Ensuite, l'existence et la position d'un contour sont estimés par une méthode de maximum de vraissemblance. Eom et Kashyap [18] estiment les paramètres d'un modèle de texture autorégressif à l'aide d'une méthode de maximum de vraissemblance. Ensuite, ils détectent la présence d'un contour par un test d'hypothèse. Manjunath et Chellappa [27] utilisent les filtres de Gabor multirésolution pour trouver les frontières entre régions d'intensité ou de texture.

Une autre façon de procéder est de détecter les contours de régions homogènes en appliquant une méthode utilisée pour les contours d'intensité sur une image de caractéristiques de texture. En effet, ces caractéristiques devraient être à peu près constantes pour une même région de texture et être différentes aux frontières [21]. Notre approche se situe dans cette optique, c'est-à-dire que nous utilisons comme caractéristiques, les moments locaux des niveaux de gris, puis nous appliquons sur les images de moments un détecteur de type gradient ou Laplacien pour trouver les frontières. Mentionnons quelques autres détecteurs qui utilisent ce genre d'approche. Khotanzad et Chen [23] ont proposé de cal-

culer les paramètres d'un modèle autorégressif. Ces paramètres sont calculés pour chaque point de l'image à partir d'un voisinage. Ensuite un filtre Sobel 3×3 est passé sur chaque image de paramètre et une combinaison des résultats les plus significatifs de ces filtrages est utilisé. Cette combinaison est la racine carrée de la somme des carrés de ces réponses. Dunn et al. [17] filtrent d'abord l'image avec un filtre de Gabor et ensuite appliquent le détecteur de contours de Canny [6]. Yhann et Young [36] commencent d'abord par étiquetter les différentes régions de texture et ensuite trouvent les frontières de ces régions à l'aide d'un détecteur de contours d'intensité.

2.3 Une nouvelle approche pour la détection de contours de texture

Comme mentionné dans la section 2.2.4, il existe des contours visibles ne correspondant pas à un point d'inflexion de l'image lissée et donc un détecteur se basant uniquement sur le niveau de gris moyen ne peut localiser ces contours. Cependant, les moments d'ordre supérieur peuvent présenter des points d'inflexion. À titre d'exemple, la moyenne des niveaux de gris de l'image de la figure 2.7a est constante (la figure 2.7b présente une détection de contours d'intensité sur 2.7a), mais la variance (figure 2.7c) présente des contours de type marche (figure 2.7d). Le deuxième moment (la variance) est particulièrement important car il caractérise le degré de contraste dans les niveaux de gris. Le troisième moment caractérise l'asymétrie de l'histogramme de la texture et le quatrième caractérise la platitude de l'histogramme [20]. Ces statistiques peuvent être calculées à partir de l'histogramme. Cependant, cette façon de faire ne tient pas compte de l'information spatiale et permet seulement d'avoir des statistiques globales ne permettant pas l'extraction des contours. Ainsi, nous les estimons sur un voisinage de pixels plutôt que sur l'image entière, de façon à ce qu'ils soient relativement constants pour une même

région de texture. Nous appelons contours de texture, les points d'inflexion de ces moments. Notre approche pour la détection de contours de texture consiste à calculer les moments locaux et à extraire les contours de ces images à l'aide d'un détecteur gradient ou Laplacien. Nous n'avons aucune préférence quant au détecteur utilisé.

Un problème sous-jacent concerne la combinaison des contours calculés à partir de chaque image de moment. Pour régler ce problème, nous nous plaçons dans le cas multibande; les moments forment un vecteur d'images (une image multibande: $\mathbb{R}^2 \Rightarrow \mathbb{R}^m$). Dans notre cas, la détection de contours de texture s'apparente donc à la détection de contours dans les images multispectrales (c.f. section 2.3.2). Le détecteur de contours multispectral prend en entrée une image multibande et fournis une seule image de contours ($\mathbf{f} : \mathbb{R}^2 \Rightarrow \mathbb{R}$).

Nous allons maintenant présenter la détection des contours par les moments et ensuite, la combinaison de ces contours.

2.3.1 Détection de contours à l'aide des moments locaux

Le calcul des moments locaux d'ordre n d'une image I au point (x, y) s'effectue de la manière suivante:

$$M_n(x, y) = E_{(x,y)}[(I(u, v) - E_{(u,v)}[I(s, t)])^n]$$

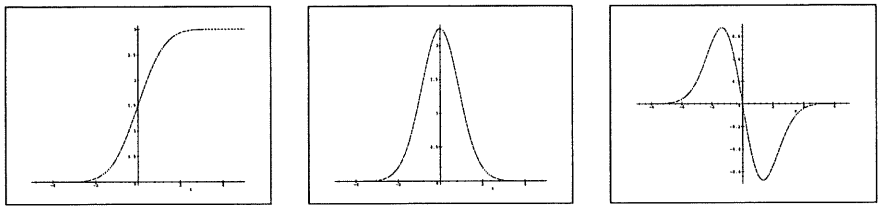
où $E_{(x,y)}[h(u, v)]$ est l'espérance d'une fonction quelconque $h(u, v)$ calculée sur un voisinage du point (x, y). Pour simplifier, $E_{(x,y)}[h(u, v)]$ peut être définie comme étant la convolution de $h(x, y)$ et de la Gaussienne d'échelle σ_m, $g_{\sigma_m}(x, y)$. Par exemple, le calcul de l'image de variance est effectué comme suit:

$$M_2(x, y) = E_{(x,y)}[I^2(u, v)] - E^2_{(x,y)}[I(u, v)] = (I^2(x, y) * g_{\sigma_m}(x, y)) - (I(x, y) * g_{\sigma_m}(x, y))^2.$$

où

$$g_{\sigma_m}(x, y) = \frac{1}{2\pi\sigma_m^2} e^{\frac{-(x^2+y^2)}{2\sigma_m^2}}$$

De cette manière, les moments sont donc calculés sur un voisinage circulaire et pondéré. Les moments d'ordre supérieur sont coûteux en temps de calcul, cependant, il existe des algorithmes pour les calculer efficacement [26]. Pour fixer les idées, la figure 2.7c présente la variance de l'image 2.7a calculée avec $\sigma_m = 1$.



(a) (b) (c)

Figure 2.2 – *(a) Contour de type marche. (b) Variance. (c) Première dérivée de la variance.*

Nous discutons maintenant du comportement de la variance et de sa dérivée, calculée à l'aide d'une convolution avec la dérivée de la Gaussienne de paramètre σ_{dm}, dans le cas de trois types de contours: marche, double marche et texture. La variance d'un contour de type marche (figure 2.2a) est un profil de ligne (figure 2.2b) et donc sa dérivée présente un passage par zéro au point de contour (figure 2.2c). Dans le cas d'un contour de type double marche (figure 2.3a), la variance présente deux maxima symétriques de chaque côté du contour lorsque σ_m est assez petit par rapport à sa largeur (figure 2.3b). Cependant, si σ_m est beaucoup plus grand, il y aura un seul maximum (figure 2.3c). La dérivée de la variance présente donc toujours un passage par zéro à l'emplacement du contour. Si σ_m est petit par rapport à σ_{dm}, ou le contraire, il n'y a qu'un seul passage par zéro (figures 2.3d et 2.3e) , tandis que si σ_m et σ_{dm} sont petits, il y aura trois passages par zéro (figure 2.3d). Pour un contour de texture (figure 2.4a), dont la moyenne des intensités est la même dans les deux régions, la variance (figure 2.4c) présente un profil de marche alors que la moyenne est presque constante (figure 2.4b).

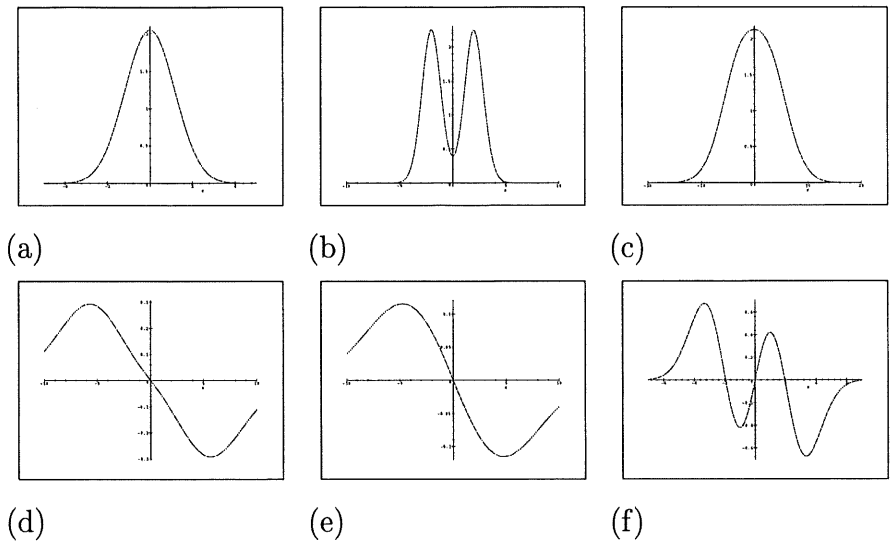
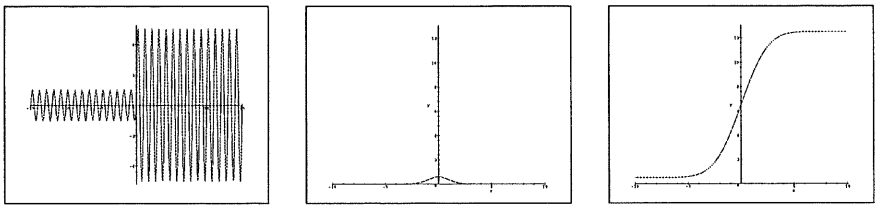


Figure 2.3 – *(a) Contour de type double marche. (b) Variance (σ_m petit). (c) Variance (σ_m grand). (d) Première dérivée de la variance (σ_m grand et σ_{dm} petit). (e) Première dérivée de la variance (σ_m petit et σ_{dm} grand). (f) Première dérivée de la variance (σ_m et σ_{dm} petits).*

2.3.2 Combinaison des types de contours

Rappelons que l'objectif est de récupérer les contours significatifs de l'image originale et des différentes images de moments. Si un tel contour est présent dans une ou plusieurs images, il devrait être présent dans l'image de contours de texture. Comme dans le cas multispectral décrit au chapitre 1, nous pouvons utiliser deux types d'approches: fusionner les contours et trouver les contours sur une image multibande. Puisque la fusion des contours est assez complexe, nous choisissons la deuxième option, l'image multibande étant formée de l'image originale et des différentes images de moment. Pour trouver les contours de cette image multibande, nous utilisons un détecteur multispectral comme celui décrit dans le chapitre 1. Cependant, il peut arriver qu'un contour soit présent dans plusieurs images, mais sous une forme différente. Par exemple, un contour de type marche dans l'image d'intensité se traduit par un contour de type double marche dans l'image de



(a) (b) (c)

Figure 2.4 – *(a) Contour de texture. (b) Moyenne. (c) Variance.*

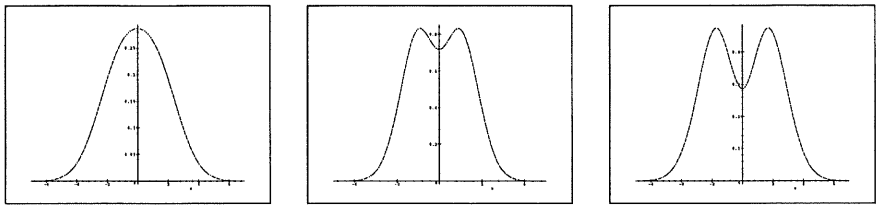
variance (figure 2.9a et 2.9c). Nous voulons que le détecteur nous donne un seul contour et non deux. Donc, dans ce cas, nous voulons privilégier le contour d'intensité plutôt que de variance en ajoutant une pondération à chaque image. Notre vecteur d'images devient alors $\mathbf{f} = (\alpha_1 I(x,y), \alpha_2 M_2(x,y), \dots)$.

Pour pouvoir utiliser un détecteur multispectral (gradient ou Laplacien), nous posons une condition, qui est que les informations dans les différentes bandes doivent être correllées. En effet, il serait inutile d'essayer d'intégrer des bandes totalement indépendantes. Comme les moments sont calculés à partir de l'image d'intensité, cette condition est respectée. De plus, dans le calcul des moments, nous avons déjà un effet de lissage puisqu'ils sont calculés à l'aide de convolutions avec une Gaussienne. Nous avons un second effet de lissage dans le calcul des dérivées partielles des moments. Pour l'image d'intensité, nous n'avons que l'effet de lissage dû au calcul des dérivées. Pour que la différence des niveaux de bruit soit réduite au maximum, il faut ajuster l'échelle utilisée pour calculer les dérivées des moments pour que le lissage total des moments soit approximativement le même que le lissage de l'image originale, c'est-à-dire, dans le cas où tous les α_i sont à 1:

$$\sigma_{di} \simeq \sqrt{\sigma_m^2 + \sigma_{dm}^2}$$

où σ_{di} est l'échelle utilisée pour calculer les dérivées de l'image originale, σ_m est l'échelle utilisée pour le calcul du moment et σ_{dm} est l'échelle utilisée pour calculer les dérivées

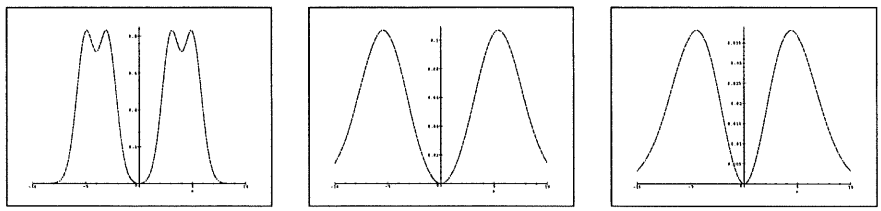
du moment.



(a) (b) (c)

Figure 2.5 – *Module du gradient multispectral sur la moyenne et la variance d'un contour de type marche. (a) σ_{dm} grand par rapport à σ_m. (b) et (c) σ_m grand par rapport à σ_{dm}.*

Pour illustrer l'effet de σ_m et de σ_{dm}, nous regardons le module du gradient multispectral pour des contours de type marche et double marche. Dans le cas d'un contour de type marche, la combinaison de la moyenne et de la variance, par le gradient multispectral, tend vers un seul maximum (figure 2.5a) si σ_{dm} augmente par rapport à σ_m et vers deux maxima symétriques de chaque côté du contour (figures 2.5b et 2.5c) quand σ_m augmente par rapport à σ_{dm}. Dans le cas d'une ligne, quand σ_m et σ_{dm} sont petits, par rapport à la largeur de la ligne, la combinaison de la moyenne et de la variance présente quatre maxima (figure 2.6a) alors que dans les autres cas, elle présente deux maxima symétriques de chaque côté de la ligne (figures 2.6b et 2.6c).



(a) (b) (c)

Figure 2.6 – *Module du gradient multispectral sur la moyenne et la variance d'un contour de type double marche. (a) σ_m et σ_{dm} sont petits. (b) σ_m grand et σ_{dm} petit. (c) σ_m petit et σ_{dm} grand.*

Nous pouvons donc résumer l'algorithme comme suit:

1. Calculer la moyenne de l'image originale.
2. Calculer les moments d'ordre supérieurs.
3. Utiliser le gradient multispectral ou le Laplacien multispectral pour trouver les contours avec une image multibande dont les différentes bandes sont les images de moyenne et des moments.

Le détecteur de contours de texture a un paramètre propre à lui, c'est-à-dire σ_m qui est le paramètre de la Gaussienne servant au calcul des moments. Dépendemment de la finesse de la texture, il peut être utile de modifier ce paramètre. Les autres paramètres sont ceux du détecteur multispectral tels que présentés au chapitre 1. σ_{dm} correspond au σ du chapitre 1.

2.4 Résultats

Nous avons testé le détecteur de textures sur des images de synthèse et des images réelles. La figure 2.7a présente une image synthétique à une bande construite de façon à ce que le niveau de gris moyen soit le même dans les trois régions alors que la variance du bruit varie. Comme l'image 2.7b le montre, une détection de contours de type gradient ($\sigma_{di} = 4.12$) ne donne pas des contours entre les régions qui sont perçues différentes par l'oeil. La figure 2.7c présente l'image de variance ($\sigma_m = 1$) obtenue à partir de l'image 2.7a. Les contours à l'échelle $\sigma_{dm} = 4$ de l'image de variance sont présentés à la figure 2.7d. Les contours des trois régions sont visibles. La figure 2.8a présente une image de plusieurs textures naturelles. La figure 2.8c présente l'image de variance calculée à l'échelle $\sigma_m = 3$. Les figures 2.8b et 2.8d sont les résultats de la détection des contours des images 2.8a et 2.8c respectivement ($\sigma_{di} = 5, \sigma_{dm} = 4$). Nous remarquons de meilleurs contours entre les

régions dans l'image 2.8d. L'image 2.8e présente le résultat de la détection de contours en utilisant à la fois l'image d'intensité et l'image de variance ($\sigma_{di} = 5, \sigma_{dm} = 4$). Pour ce faire, nous avons utilisé le détecteur gradient multispectral. Sur cette image, le contour de la zone de droite est mieux fermé que dans 2.8d et le contour du coin supérieur gauche de 2.8b est conservé. En revanche, une partie du contour de la région inférieure de 2.8d est perdue. Nous n'avons pas cherché la pondération optimale des deux images. En effet, nous avons utilisé $\alpha_i = 1, \forall i$. La figure 2.9 présente le résultat d'une détection de type Laplacien sur l'image 1.6b. Cette image présente un contour de variance à l'intérieur du rectangle en bas à gauche. Comme on le voit sur la figure 2.9c, ce contour n'apparait pas. Par contre, il apparait clairement sur les figures 2.9b et 2.9d qui sont respectivement l'image de variance ($\sigma_m = 1$) et ses passages par zéro du Laplacien ($\sigma_{dm} = 2.83$). Les contours des autres rectangles sont soit doubles ou disparus. La figure 2.9d présente les passages par zéro du Laplacien multibande développé à la section 1.3 ($\sigma_{di} = 3, \sigma_{dm} = 2.83$). Les contours de type marche des deux images sont conservés.

2.5 Conclusion

Dans ce chapitre, nous avons présenté un aperçu de l'analyse de texture ainsi qu'une nouvelle approche pour la détection de contours de textures. Nous utilisons les moments locaux des niveaux de gris pour caractériser les textures puis nous effectuons une détection de contours multispectrale sur une image multibande formée de l'image d'intensité et des images des différents moments. Nous avons présenté quelques résultats utilisant la moyenne et la variance. Il faudrait étudier plus en profondeur l'impact des moments d'ordre supérieurs à 2. Il serait aussi intéressant d'effectuer une étude plus approfondie de l'effet de l'échelle. De plus, il serait intéressant de vérifier la pertinence d'une pondération des différentes images, par exemple pour augmenter l'impact de l'intensité par rapport à

la variance ou vice-versa.

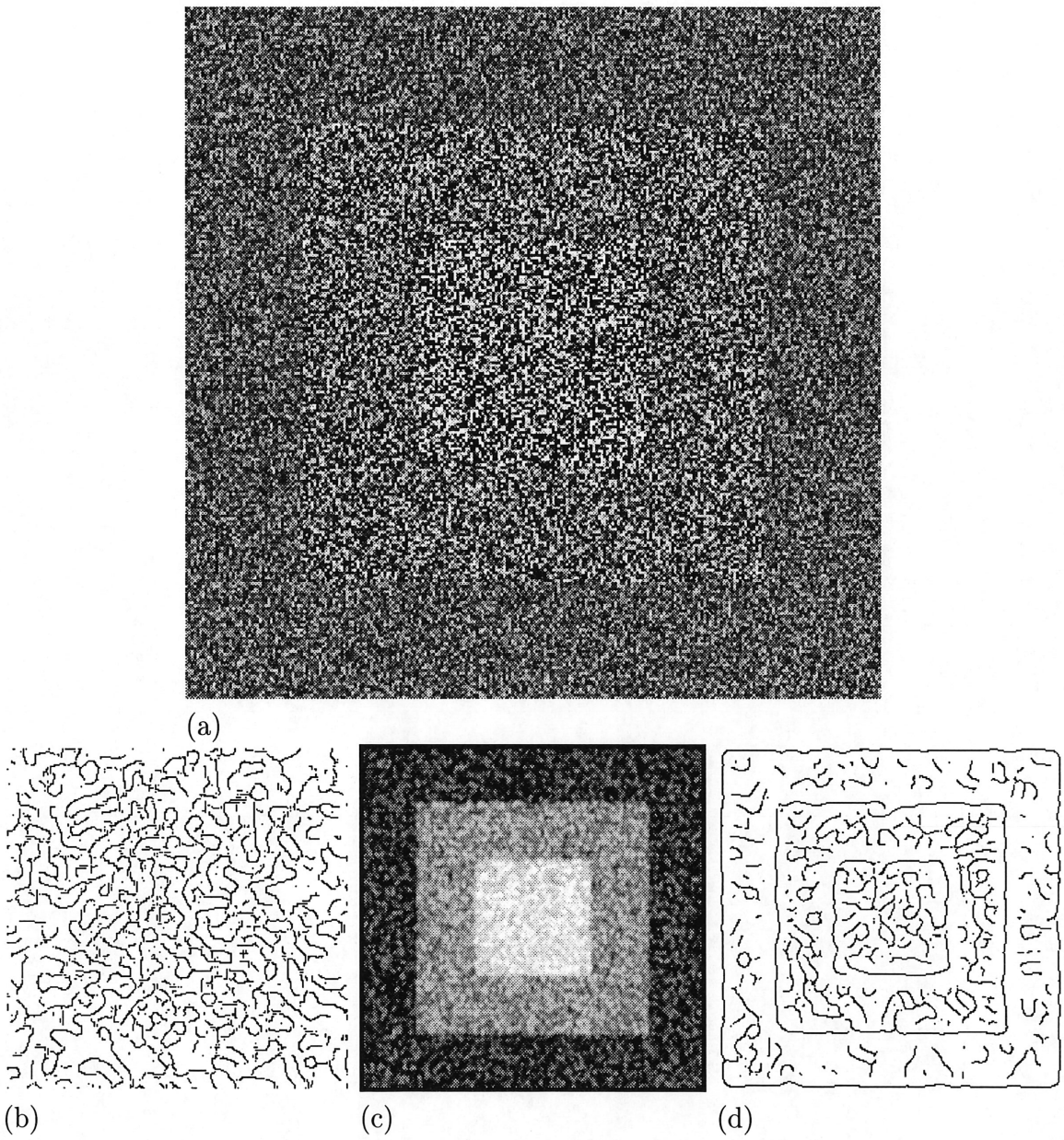


Figure 2.7 – *(a) Image synthétique. Les niveaux de gris moyens dans les trois régions sont les mêmes. Cette image est affichée à une plus grande résolution de façon à rendre plus visible les différentes régions. (b) Détection de contours par le gradient de la Gaussienne ($\sigma_{di} = 4.12$). (c) Image de variance ($\sigma_m = 1$) avec égalisation d'histogramme. (d) Contours de l'image de variance obtenus par le gradient de la Gaussienne ($\sigma_{dm} = 4$).*

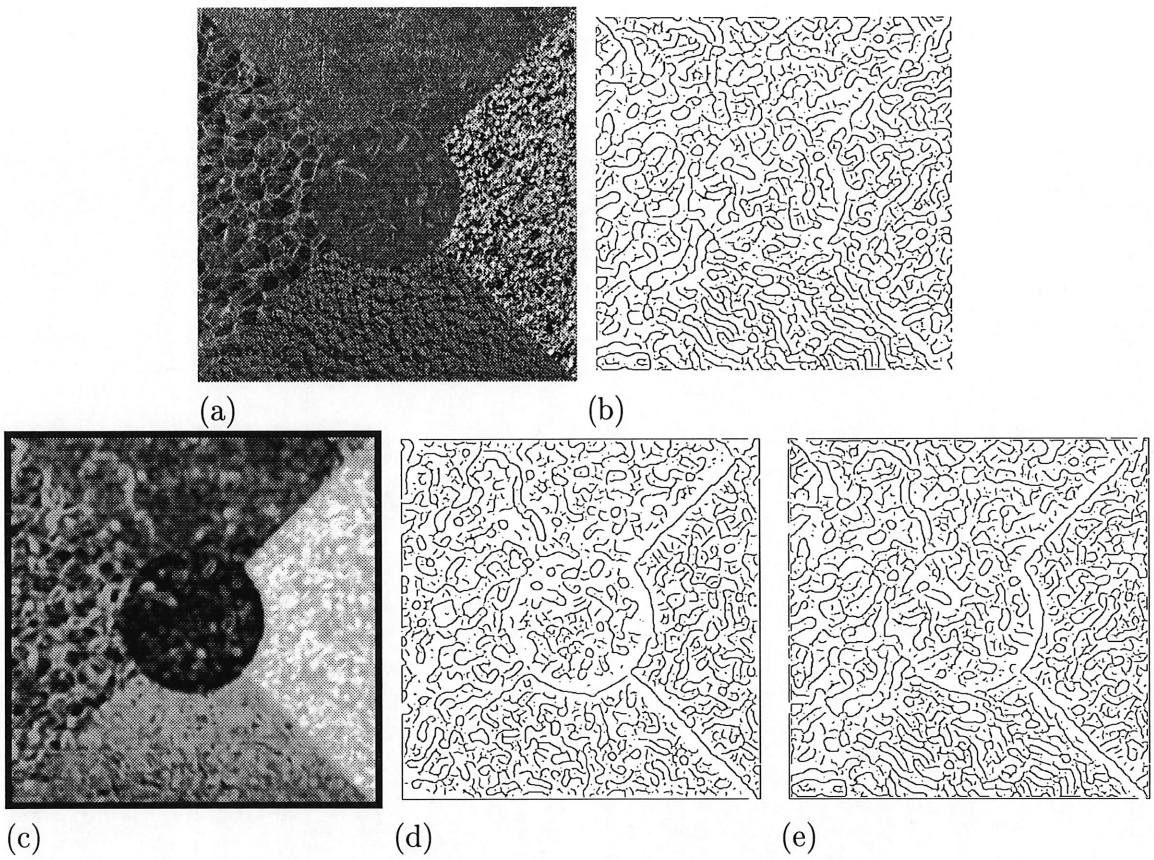


Figure 2.8 – *(a) Mixture de textures de Brodatz. (b) Contours obtenus par un gradient de la Gaussienne ($\sigma_{di} = 5$). (c) Image de variance ($\sigma_m = 3$) avec égalisation d'histogramme. (d) Contours de l'image de variance obtenus par gradient de la Gaussienne ($\sigma_{dm} = 4$). (e) Contours de l'image originale et de l'image de variance par le gradient multispectral ($\sigma_{di} = 5$, $\sigma_{dm} = 4$).*

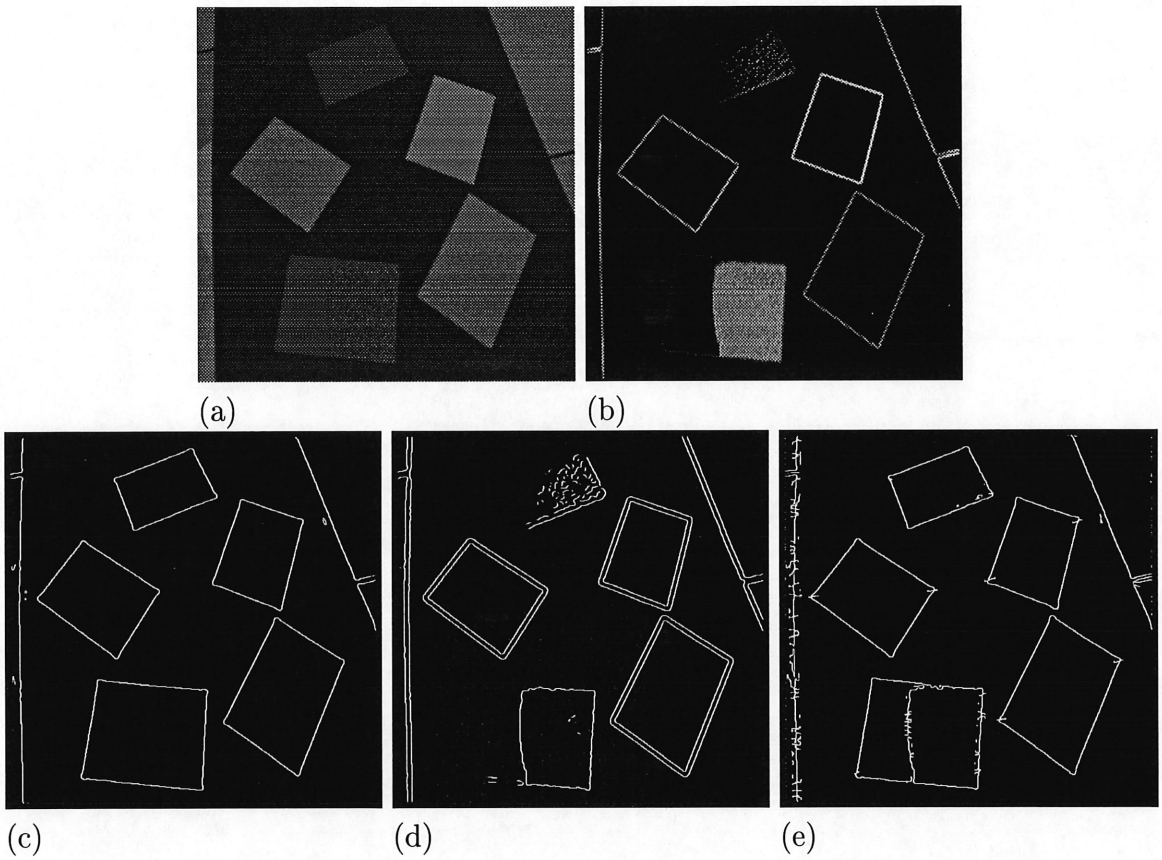


Figure 2.9 – *(a) Bande verte de l'image de papier (b) Image de variance de a ($\sigma_m = 1$). (c) Contours de l'image a obtenus par le Laplacien de la Gaussienne ($\sigma_{di} = 3$). (d) Contours de l'image de variance obtenus par le Laplacien de la Gaussienne ($\sigma_{dm} = 2.83$). (e) Contours de l'image originale et de l'image de variance obtenus par le Laplacien multispectral (non-corrigé) ($\sigma_{di} = 3$ et $\sigma_{dm} = 2.83$).*

CHAPITRE 3

DÉTECTION DE ROUTES DANS LES IMAGES À HAUTE RÉSOLUTION

3.1 Introduction

La détection des routes est une opération très longue lorsqu'effectuée manuellement. L'imagerie aérienne et satellite peut faciliter ce processus mais l'automatisation complète est encore un but lointain. La détection de routes est un type d'extraction de caractéristiques, comme la détection de contours [37, 4], qui est généralement dépendante du contexte. Dans notre cas, nous avons accès à des images à haute résolution géoréférencées, c'est-à-dire que nous connaissons la position géographique 2D de chaque pixel, et à des informations sur la localisation des routes qui nous sont fournies par Géomatique Canada, ces informations étant aussi géoréférencées. Les informations de route sont obtenues à partir de la numérisation des cartes routières existantes et donc elles présentent

Figure 3.1 – *Exemple de l'imprécision de la base de données de routes. Les lignes blanches sont les routes tirées de la base de données fournie par Géomatique Canada.*

deux problèmes: 1) une précision inégale (figure 3.1) et 2) les nouvelles routes ne sont pas incluses. Le but de notre travail est donc la mise à jour automatique de ces informations.

Plusieurs travaux ont été faits pour automatiser la détection de routes [3]. Parmis les méthodes développées, nous pouvons mentionner particulièrement le travail de Klang [24]. Il a développé une méthode pour détecter les changements entre une base de données de routes et une image satellite. D'abord, il utilise les routes de la base de données pour initialiser un processus d'optimisation par contours actifs dans le but de corriger l'information existante. Ensuite, il effectue un suivi de ligne avec une approche statistique pour détecter les nouvelles routes à partir du réseau existant. Nous avons constaté qu'aucune des méthodes passées en revue n'utilisait l'information de jonctions de lignes de l'image. Ces jonctions sont généralement fiables et comme les routes sont construites en réseaux, elles sont pertinentes dans le contexte de leur détection. Notre approche pour la correction des informations connues est similaire à celle de Klang, mais nous y ajoutons les jonctions de lignes. Pour la détection de nouvelles routes, nous débutons le suivi de lignes à partir des jonctions de lignes près du réseau connu. Nous décrivons notre approche dans la section 3.2. Dans la section 3.3, nous présentons les détecteurs

de lignes et de jonctions de lignes. Dans la section 3.4, nous décrivons la correction des informations de routes déjà connues. Dans la section 3.5, nous présentons la détection des nouvelles routes. Dans la section 3.6, nous présentons les résultats obtenus avec une image géoréférencée et une base de données de routes, fournies par Géomatique Canada. Finalement, nous présentons les perspectives pour le projet ainsi que la conclusion dans la section 3.7.

3.2 Notre approche

La correction de la base de données est la première étape pour la mise à jour des cartes routières. Dû au fait que ces bases de données ont d'abord été obtenues en numérisant les cartes routières existantes, la localisation des routes n'est pas précise. Cependant cette information permet de distinguer les éléments linéaires qui sont des routes et ceux qui ne le sont pas. Dans les images aériennes ou satellites (i.e. à haute résolution), les routes sont généralement des caractéristiques longues et minces. Nous voulons donc trouver les lignes qui sont proches de l'initialisation fournie par la base de données. Cela nous amène naturellement vers les contours actifs qui sont des courbes d'optimisation qui requièrent une initialisation proche de la solution. La fonction à optimiser est une fonction des lignes rehaussées de l'image. L'initialisation de la base de données n'est cependant pas nécéssairement dans la zone d'attraction du contour, c'est-à-dire dans un voisinage proche du maximum de la ligne. Pour rapprocher cette initialisation de la solution, nous effectuons une mise en correspondance entre chaque intersection de route de la base de données et, au plus, une jonction de lignes de l'image. Nous effectuons alors un repositionnement de chaque segment de route par rapport à la différence entre les intersections et les jonctions. Après cette relocalisation, nous appliquons une optimisation de l'image de lignes rehaussées par contours actifs.

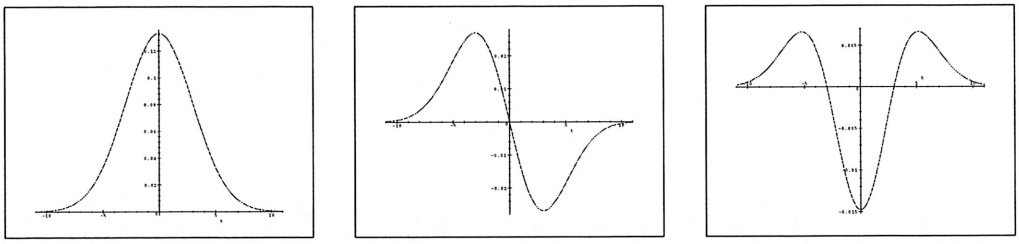
La deuxième étape est la génération d'hypothèses pour les nouvelles routes. Nous savons que les routes sont construites en réseaux et que les nouvelles sont liées avec les anciennes. Nous générons des hypothèses pour les nouvelles routes en effectuant un suivi des lignes qui partent des jonctions de lignes situées près du réseau connu.

L'algorithme de mise à jour peut être résumé en six étapes:

1. Réduction de la résolution de l'image en la rééchantillonant, dans le cas où elle n'est pas assez basse pour la détection de lignes.
2. Extraction des lignes en utilisant l'algorithme de Steger [31] qui sera présenté dans la section 3.3.
3. Recherche des jonctions de lignes avec le résultat de la détection de lignes (section 3.3.1).
4. Mise en correspondance des extrémités des segments de route avec les jonctions de lignes et déplacement des segments de route selon la différence entre les extrémités et les jonctions (section 3.4.1).
5. Relocalisation de chaque segment de route, un à un, avec l'approche par contours actifs.
6. Suivi des lignes partant des jonctions près du réseau existant, dans le but de générer des hypothèses pour les nouvelles routes.

3.3 Détection de lignes et de jonctions de lignes

Dans les images à haute résolution, les routes sont des éléments longs et minces. Habituellements, elles sont des lignes claires sur un fond plus foncé. Pour les détecter, nous utilisons une détection de lignes. Nous présentons dans cette section le détecteur proposé par Steger [31].



(a) (b) (c)

Figure 3.2 – *Coupe d'une fonction de ligne. (a) Profil d'une ligne. (b) Première dérivée. (c) Seconde dérivée.*

Une ligne claire est un maximum dans l'image à niveaux de gris (figure 3.2a). Plusieurs algorithmes de détection de lignes ont été proposés [37]. L'idée de base derrière celui que nous utilisons est que la deuxième dérivée d'une ligne claire dans sa direction perpendiculaire présente un minimum négatif (figure 3.2c) et sa première dérivée est un passage par zéro (figure 3.2b).

Dans le cas 2D, nous devons d'abord trouver la direction perpendiculaire à la ligne, c'est-à-dire présentant la courbure maximale de la fonction. Posons $f(x, y)$, une image discrète. Cette direction peut être déterminée en calculant les valeurs propres et les vecteurs propres de la matrice du Hessian:

$$H(x, y) = \begin{bmatrix} f_{xx} & f_{xy} \\ f_{xy} & f_{yy} \end{bmatrix},$$

où f_{xx}, f_{xy} et f_{yy} sont les dérivées partielles de second ordre de $f(x, y)$.

La direction présentant la courbure maximale est la direction perpendiculaire à la ligne. C'est la direction du vecteur propre correspondant à la valeur propre ayant la valeur absolue maximale. Si cette valeur propre est négative, c'est une ligne claire et si elle est positive, c'est une ligne foncée. Les deux valeurs propres de $H(x, y)$ peuvent être calculées

facilement par:

$$\lambda = \frac{(f_{xx} + f_{yy}) \pm \sqrt{(f_{xx} - f_{yy})^2 + 4f_{xy}^2}}{2}.$$

La direction normalisée perpendiculaire à la ligne est

$$\vec{n} = (n_x, n_y) = \begin{cases} \dfrac{(\lambda_{max} - f_{yy}, f_{xy})}{\sqrt{(\lambda_{max} - f_{yy})^2 + f_{xy}^2}} & \text{si } f_{xy} = \lambda_{max} - f_{xx} = 0 \\ \dfrac{(f_{xy}, \lambda_{max} - f_{xx})}{\sqrt{f_{xy}^2 + (\lambda_{max} - f_{xx})^2}} & \text{ailleurs} \end{cases} \tag{3.1}$$

où λ_{max} est la valeur propre ayant la valeur absolue maximale au point (x, y). Avec cette valeur propre, nous pouvons définir une mesure de plausibilité de ligne. C'est-à-dire que lorsque nous cherchons les lignes foncées, la plausibilité de ligne est définie comme étant λ_{max} et lorsque nous cherchons les lignes claires, la plausibilité de ligne est définie comme étant $-\lambda_{max}$.

La ligne a un profil unidimensionnel dans la direction $\vec{n}$, alors développons l'approche 1D dans cette direction. Posons $f(x)$, une image discrète. Pour déterminer l'emplacement de la ligne, nous approximons l'image par le polynôme de deuxième ordre de Taylor, généré au point x:

$$p_2(s) = f(x) + f'(x)s + \frac{1}{2}f''(x)s^2$$

où $s \in \mathbb{R}$, $f'(x)$ et $f''(x)$ sont les premières et deuxièmes dérivées de $f(x)$. La position de la ligne est le point où $p_2'(s) = 0$:

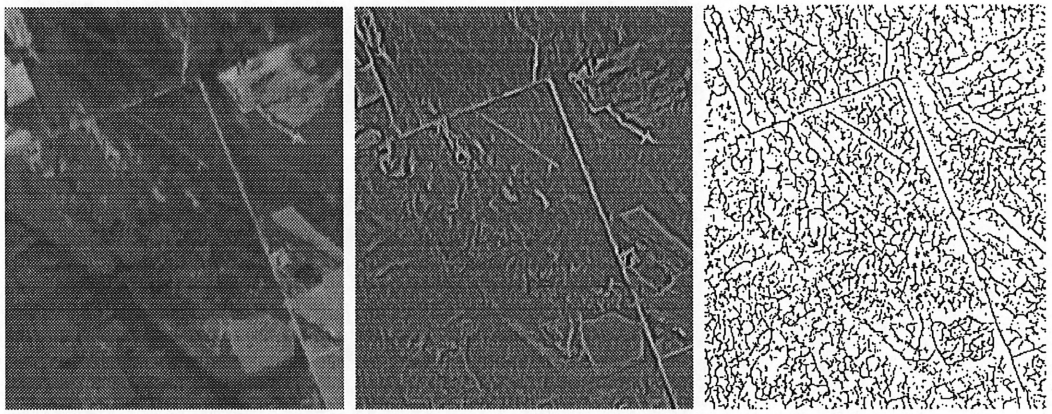
$$s = -\frac{f'(x)}{f''(x)}.$$

Quand s est dans les bornes du pixel ($s \in [-1/2, 1/2]$) et que $|f''(x)|$ est plus grand qu'un certain seuil, x est considéré comme un point de ligne. Si $f''(x) > 0$, c'est une ligne

foncée et si $f''(x) < 0$, c'est une ligne claire. En 2D, nous avons donc

$$\vec{p} = \left(-\frac{\frac{\partial f(x,y)}{\partial \vec{n}}}{\frac{\partial^2 f(x,y)}{\partial \vec{n}^2}} \right) \vec{n} = \left(-\frac{f_x n_x + f_y n_y}{f_{xx} n_x^2 + 2 f_{xy} n_x n_y + f_{yy} n_y^2} \right) \vec{n},$$

où f_x et f_y sont les dérivées partielles de premier ordre de $f(x, y)$. Un pixel (x, y) est un point de ligne si $\vec{p} \in [-1/2, 1/2] \times [-1/2, 1/2]$ et λ_{max} est utilisé pour sélectionner les lignes claires évidentes. Quand la largeur de la ligne est plus d'un pixel, nous pouvons utiliser une suppression des non-minima sur λ_{max}, dans la direction $\vec{n}$. Pour le calcul des dérivées, nous effectuons des convolutions avec les dérivées de la Gaussienne de paramètre σ. La figure 3.3 présente une image, sa plausibilité de ligne claire et les lignes obtenues avec $\sigma = 1$.



(a) (b) (c)

Figure 3.3 – *Détection de lignes. a) Image originale. b) Plausibilité des lignes claires ($-\lambda_{max}$) ($\sigma = 1$). c) Lignes détectées.*

3.3.1 Jonctions de lignes

Dans le contexte de la détection de routes à partir d'images à haute résolution, nous définissons les jonctions comme les intersections des lignes. Les jonctions sont des caractéristiques très utiles, particulièrement pour les problèmes de mise en correspondance,

car elles sont des éléments stables et fiables. Malheureusement, nous remarquons qu'elles ne sont pas utilisées dans la détection de routes. Une raison à cette lacune est qu'à notre connaissance, avant d'entamer ce projet, il n'existait aucun détecteur de jonctions de lignes. Nous utilisons un tel détecteur, développé dans notre groupe de recherche parallèlement à ce projet, qui est basé sur une mesure de courbure entre les vecteurs directionnels des pixels de ligne situés dans un voisinage donné. La jonction est localisée dans une région de forte courbure. Pour plus de détails, le lecteur intéressé peut consulter le rapport de recherche sur ce détecteur [12]. La figure 3.4 montre un résultat du détecteur de jonctions de lignes.

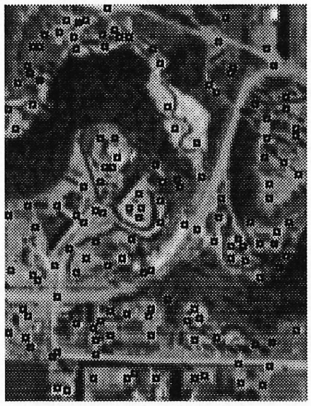


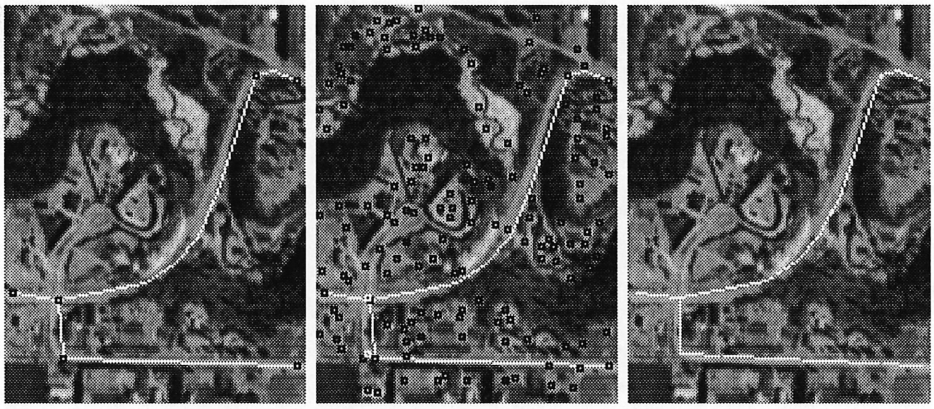
Figure 3.4 – *Jonctions de lignes (carrés noirs).*

3.4 Correction des informations de route

La correction des informations de route comporte deux étapes. La première est une relocalisation des routes connues pour les rapprocher de la solution finale du contour actif. La deuxième étape est le processus du contour actif lui-même. Ces étapes sont présentées dans les deux prochaines sections.

3.4.1 Mise en correspondance entre la base de données et l'image

Le contour actif est un processus itératif nécessitant une initialisation qui soit proche de la solution finale. Cette initialisation doit être à l'intérieur de la zone d'attraction du contour. Nous proposons d'utiliser les jonctions de lignes pour relocaliser l'information de la base de données de routes, car elles indiquent l'endroit dans l'image où se trouvent les intersections de routes. À cette fin, nous utilisons une mesure de corrélation entre quelques points appartenant à la base de données et les jonctions de lignes.



(a) (b) (c)

Figure 3.5 – *Mise en correspondance entre la base de données et l'image. a) Segments de route. Les carrés noirs sont les extrémités. b) Mise en correspondance entre les extrémités de segments de route et les jonctions de lignes. Les carrés blancs sont les jonctions correspondantes aux extrémités tandis que les carrés noirs sont, soit les extrémités, soit les autres jonctions de lignes. Le voisinage est* (N_m) 13×13. *c) Segments de route après la relocalisation.*

La base de données fournie par Géomatique Canada est construite en liste de points géoréférencés, définissant des segments de ligne. Nous appellons ces segments de ligne, vecteurs. Ces points étant en coordonnées géoréférencées, il nous faut d'abord les convertir en coordonnées relatives au début de l'image. Ensuite, nous regroupons les vecteurs en segments de route dans le but de réduire le nombre de contours actifs. Ces segments de route doivent être les plus droits possible de façon à pouvoir ajouter une contrainte

de grande rigidité au contour actif. Cette grande rigidité aide à éviter le bruit et les problèmes d'ombrage car le contour optimisé a tendance à moins suivre les petites variations d'intensité causées par ces phénomènes. Nous lions deux vecteurs s'ils sont adjacents, si le cosinus de l'angle entre eux est plus petit qu'un certain seuil (dans nos test, nous avons mis ce seuil à $\cos 138°$ après quelques essais) et s'il n'y a aucun autre vecteur partant de l'extrémité commune aux deux vecteurs. Donc, nous trouvons les extrémités de segments de route, premièrement aux endroits où plus de deux vecteurs aboutissent au même point, deuxièmement aux endroits où l'angle entre deux vecteurs est plus petit qu'une valeur et troisièmement aux terminaisons. Un segment de route est décrit par tous les vecteurs entre ses deux extrémités. La figure 3.5a montre cinq segments de route.

Nous avons donc la liste de tous les segments de route. Nous voulons mettre en correspondance, si possible, chaque extrémité avec une jonction. Pour cela, nous appliquons une mesure de corrélation entre l'extrémité et chaque jonction dans un voisinage $N_m \times N_m$ autour de l'extrémité. Cette mesure est effectuée dans un voisinage $M_m \times M_m$ autour de la jonction. Posons l'extrémité (x, y) et la jonction (u, v), la mesure de corrélation est:

$$corr(x, y, u, v) = \sum_{i=-M_m/2}^{M_m/2} \sum_{j=-M_m/2}^{M_m/2} SegRoute(x+i, y+j) line(u+i, v+j), \qquad (3.2)$$

où

$$SegRoute(x, y) = \begin{cases} 1 & \text{si } (x, y) \text{ appartient à un segment de route} \\ 0 & \text{sinon.} \end{cases}$$

Nous mettons en correspondance (x, y) avec la jonction (u, v) ayant la plus grande corrélation. La figure 3.6 présente un exemple du choix des jonctions pour le calcul de corrélation. La corrélation est calculée pour les jonctions a, b et c, mais non pour d et e puisqu'elles sont en dehors du voisinage déterminé par N_m. La figure 3.5b présente le résultat de la mise en correspondance entre les extrémités présentées dans la figure 3.5a et les jonctions de la figure 3.4. Nous avons utilisé un M_m de 25 dans tous nos tests.

Après la mise en correspondance, nous relocalisons chaque segment de route par rapport

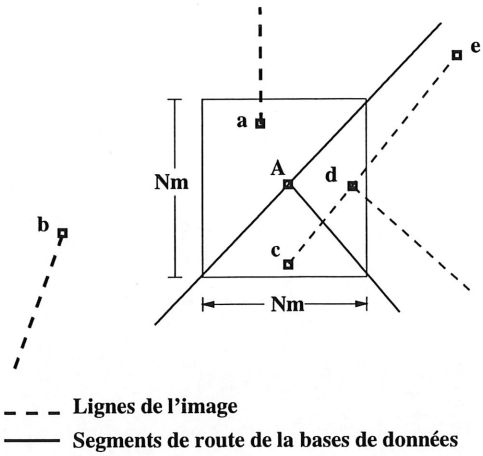


Figure 3.6 – *Choix des jonctions pour le calcul de la corrélation. A est l'extrémité du segment de route pour lequel on veut trouver la meilleure jonction. a, b, c, d et e sont les jonctions de lignes de l'image. La corrélation est calculée pour les jonctions a, b et c, mais non pour d et e*

à la différence entre ses extrémités et les jonctions correspondantes. Comme la différence peut ne pas être la même aux deux extrémités, nous déplaçons chaque extrémité du vecteur de la différence entre l'extrémité de segment de route la plus proche et sa jonction correspondante. C'est-à-dire que chaque vecteur garde la même orientation sauf celui contenant le point milieu du segment de route car ses deux extrémités seront déplacées par rapport aux deux extrémités du segment (figure 3.7). La figure 3.5c présente les segments de route après la relocalisation.

3.4.2 Contours actifs

Les routes peuvent être cachées partiellement par des ombrages d'arbres, de maisons ou de nuages et leur détection peut être affectée par le bruit. Les contours actifs sont adaptés pour corriger la base de données car ils ont une contrainte de rigidité permettant de minimiser ces problèmes.

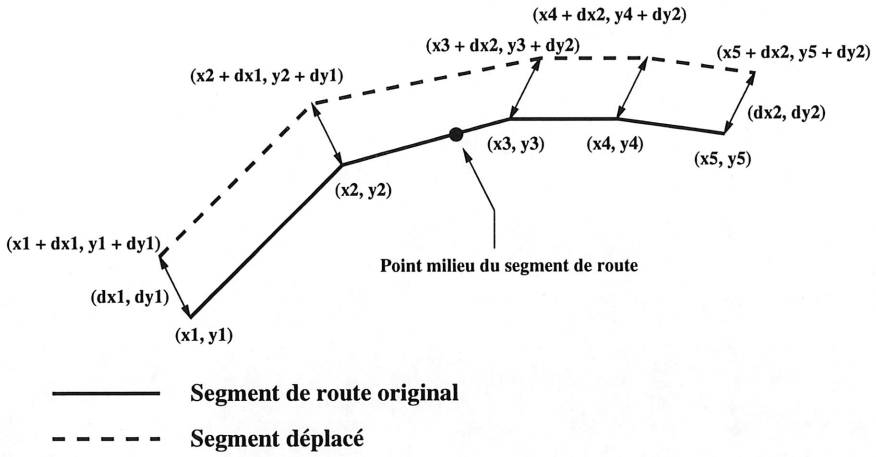


Figure 3.7 – *Exemple du déplacement d'un segment de route après la mise en correspondance.* $(u_1, v_1) = (x_1 + d_{x_1}, y_1 + d_{y_1})$ *est la jonction mise en correspondance avec l'extrémité* (x_1, y_1) *du segment de route original.* $(u_2, v_2) = (x_5 + d_{x_2}, y_5 + d_{y_2})$ *est la jonction mise en correspondance avec l'extrémité* (x_5, y_5) *du segment de route original.*

Dans le domaine continu, les contours actifs sont des courbes paramétriques $(v(s,t) = (x(s,t), y(s,t), s \in [0,1])$ qui minimisent une fonction d'énergie basée sur des contraintes internes et externes [5] au temps t:

$$E_{tot} = E_{ext} + \eta E_{int} = \int_0^1 \left(E_{ext}(v(s,t)) + \eta E_{int}(v(s,t))\right) ds \tag{3.3}$$

où E_{ext} est l'énergie externe, E_{int} l'énergie interne et η est un paramètre de régularisation.

L'énergie interne représente la capacité du contour à se déformer:

$$E_{int} = \int_0^1 \left(\alpha_1 |v'(s,t)|^2 + \beta_1 |v''(s,t)|^2\right) ds, \tag{3.4}$$

où $\alpha_1, \beta_1 \in \mathbb{R}^+$, v' et v'' sont les première et deuxième dérivées partielles en s. Le terme $|v'(s,t)|^2$ influence la longueur du contour (ou la tension) et $|v''(s,t)|^2$ influence la courbure (ou l'élasticité) du contour. Le paramètre η peut être absorbé dans α et β ($\alpha = \eta\alpha_1$ et $\beta = \eta\beta_1$).

L'énergie externe représente la force de l'image $(f(x,y))$ attirant le contour. Nous voulons maximiser cette force, ce qui revient à minimiser son inverse négatif. Dans notre cas, cette

force est la plausibilité de ligne claire ($f(x,y) = -\lambda_{max}(x,y)$) telle que calculée par le détecteur de lignes:

$$E_{ext} = \int_0^1 -f(v(s,t) = \int_0^1 \lambda_{max}(v(s,t))ds. \tag{3.5}$$

L'équation 3.3 est un problème d'évolution (la position du contour actif évolue dans le temps). La solution à ce problème de minimisation est l'équation d'évolution d'Euler simplifiée:

$$\gamma\frac{\partial v(s,t)}{\partial t} - \alpha v''(s,t) + \beta v''''(s,t) = -\frac{\partial E_{ext}(v(s,t))}{\partial v}, \tag{3.6}$$

où v'''' est la quatrième dérivée partielle en s. Le paramètre γ représente la viscosité de la courbe. Plus cette viscosité est grande, plus il y a de l'inertie dans la courbe et plus son évolution est lente.

La version discrète de l'équation 3.6 est obtenue en posant une version discrète de la courbe paramétrique au moment t: $\mathbf{V}_t = \{v_{i,t} = (x_{i,t}, y_{i,t}), i = 1..n\}$ où n est le nombre de points dans la courbe. Nous pouvons approximer la première dérivée par:

$$v'_{i,t} = v_{i,t} - v_{i-1,t},$$

et la deuxième dérivée par:

$$v''_{i,t} = v_{i+1,t} - 2v_{i,t} + v_{i-1,t}.$$

La dérivée temporelle est approximée par:

$$\frac{\partial v_{i,t}}{\partial t} = v_{i,t} - v_{i,t-1}.$$

La solution peut donc être réécrite sous la forme vectorielle suivante:

$$\gamma\frac{\partial \mathbf{V}_t}{\partial t} - \alpha\mathbf{V}''_t + \beta\mathbf{V}''''_t = -\frac{\partial \mathbf{E}_{ext}(\mathbf{V}_{t-1})}{\partial v}. \tag{3.7}$$

Au point $v_{i,t}$, nous avons:

$$\gamma(v_{i,t} - v_{i,t-1}) - \alpha(v_{i+1,t} - 2v_{i,t} + v_{i-1,t}) + \beta(v_{i+2,t} - 4v_{i+1,t} + 6v_{i,t} - 4v_{i-1,t} + v_{i-2,t}) = -\frac{\partial E_{ext}(v_{i,t-1})}{\partial v} \tag{3.8}$$

Nous pouvons réécrire l'équation 3.7 sous la forme matricielle:

$$(\mathbf{K} + \gamma\mathbf{I})\mathbf{V}_t = \gamma\mathbf{V}_{t-1} - \frac{\partial \mathbf{E}_{ext}(\mathbf{V}_{t-1})}{\partial v}, \tag{3.9}$$

où I est la matrice identité $n \times n$ et $\mathbf{K}$ est une matrice $n \times n$. Les éléments de $\mathbf{K}$ sont déterminés par l'équation 3.8 et par les conditions initiales; le contour a des extrémités libres, c'est-à-dire $v''(0,t) = v'''(0,t) = v''(1,t) = v'''(1,t) = 0$. Donc $\mathbf{K}$ est définie par:

$$\begin{bmatrix}
\beta & -2\beta & \beta & 0 & \dots & \dots & \dots & \dots & \dots & 0 \\
-\alpha-2\beta & 2\alpha+5\beta & -\alpha-4\beta & \beta & 0 & \dots & \dots & \dots & \dots & 0 \\
\beta & -\alpha-4\beta & 2\alpha+6\beta & -\alpha-4\beta & \beta & 0 & \dots & \dots & \dots & 0 \\
0 & \beta & -\alpha-4\beta & 2\alpha+6\beta & -\alpha-4\beta & \beta & 0 & \dots & \dots & 0 \\
\vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & & & \vdots \\
\vdots & & & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots \\
0 & \dots & \dots & 0 & \beta & -\alpha-4\beta & 2\alpha+6\beta & -\alpha-4\beta & \beta & 0 \\
0 & \dots & \dots & \dots & 0 & \beta & -\alpha-4\beta & 2\alpha+6\beta & -\alpha-4\beta & \beta \\
0 & \dots & \dots & \dots & \dots & 0 & \beta & -\alpha-4\beta & 2\alpha+6\beta & -\alpha-4\beta \\
0 & \dots & \dots & \dots & \dots & \dots & 0 & \beta & 2\beta & \beta
\end{bmatrix}$$

γ est donné par [5]:

$$\gamma = \frac{|\frac{\partial \mathbf{E}_{tot}}{\partial v}|}{\sqrt{n}\Delta},$$

où Δ est la progression moyenne de la courbe. Le Δ initial doit être déterminé par l'usager. Nous gardons le même Δ jusqu'à ce que l'énergie totale de la courbe augmente et alors nous diminuons Δ par un facteur de 0.75. Nous pouvons approximer l'énergie totale de la courbe au temps t par:

$$E_{tot,t} = \sum_{i=1}^{n} \left(E_{ext}(v_{i,t}) + \alpha|v'_{i,t}|^2 + \beta|v''_{i,t}|^2\right)$$

Pour résumer l'étape de correction, pour chaque segment de route, nous effectuons les tâches suivantes:

1. Initialiser un contour actif ($\mathbf{V}_0$) avec le segment de route relocalisé résultant de

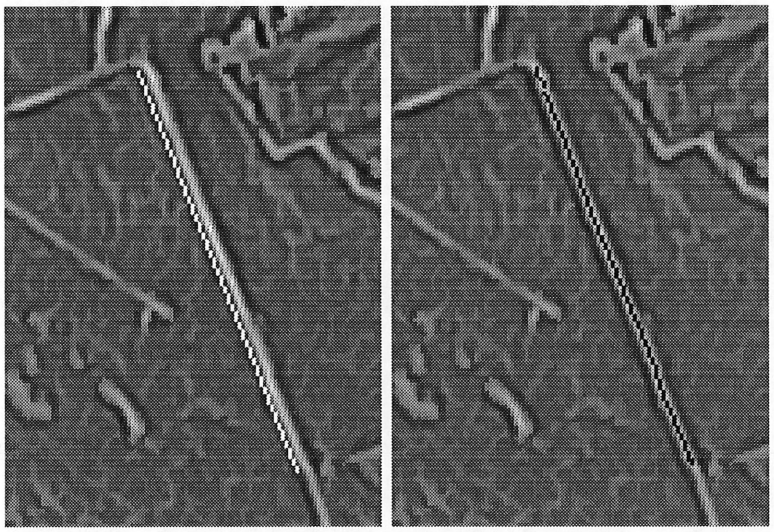
l'étape de mise en correspondance. Chaque pixel appartenant à la courbe est un $v_{i,0}$. Poser t à 1.

2. Calculer $\mathbf{V}_t$ à partir de l'équation 3.9.

3. Si l'énergie totale de la courbe diminue, alors incrémenter t et aller à l'étape 2.

4. Si Δ n'est pas trop bas, diminuer Δ, incrémenter t et aller à l'étape 2.

La figure 3.8 montre un exemple de contour actif utilisé pour optimiser une fonction de plausibilité de ligne ($\alpha = 100$ and $\beta = 0$). Pour tous les tests, nous avons utilisé un Δ initial de 1. Nous voyons que même si l'initialisation est clairement à côté de la ligne, après 18 itérations, le contour est au centre de la ligne.



(a) (b)

Figure 3.8 – *Contours actifs. a) Initialisation. Chaque pixel est un point de* V_0. *b) Après* 18 *itérations.*

3.5 Génération d'hypothèses pour les nouvelles routes

Comme mentionné plus tôt, le deuxième problème pour la mise à jour des informations routières concerne la détection des nouvelles routes. Ces nouvelles routes sont aussi des lignes claires sur fond foncé, donc le résultat de la détection de lignes peut être utilisé.

Les routes sont construites pour permettre aux gens de voyager d'un endroit à l'autre. Cela n'aurait donc aucun sens d'avoir un segment de route qui ne serait pas relié avec le réseau routier existant. Pour cette raison, nous considérons seulement les lignes qui sont connectées au réseau de la base de données existante. L'algorithme de suivi de ligne débute à partir des jonctions dans le voisinage $(N \times N)$ des routes corrigées par contours actifs (figure 3.9).

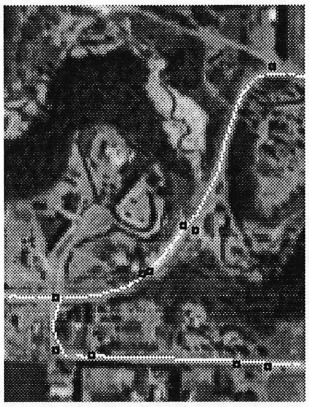


Figure 3.9 – *Jonctions de lignes près des routes corrigées* $(N = 7)$.

Nous suivons toutes les lignes partant de ces jonctions. Nous considérons chaque pixel de ligne, dans le voisinage d'une jonction (5×5), comme un pixel de départ pour le suivi de ligne (figure 3.10).

Pour chaque ligne, nous procédons comme suit: initialement, nous prennons un des pixels de départ (n'importe lequel). Posons p, le dernier pixel retenu (nous l'appelons parent). D'abord, nous considérons trois voisins de ce pixel dans la direction de la ligne. Par

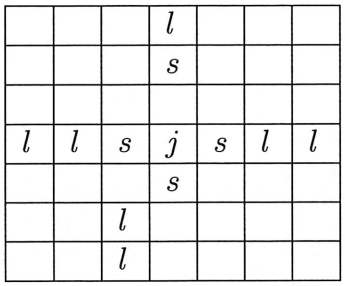


Figure 3.10 – *Une jonction de lignes près du réseau existant (j), ses pixels de départ (s) et les pixels de ligne (l).*

exemple, si la direction de la ligne au point $p = (p_x, p_y)$ est dans l'intervalle $]\pi/8, 3\pi/8]$, les trois voisins considérés sont $(p_x + 1, p_y + 1)$, $(p_x, p_y + 1)$ et $(p_x + 1, p_y)$ (voir la table 3.1). La direction de la ligne en p est calculée à partir de $\vec{n}$ (équation 3.1). Chacun de ces trois voisins qui est un pixel de ligne est considéré comme enfant. Le prochain pixel retenu est l'enfant minimisant:

$$|\theta_c - \overline{\theta}_p| \tag{3.10}$$

où θ_c est la direction de ligne de l'enfant et $\overline{\theta}_p$ est la direction moyenne de ligne du parent. La direction moyenne est calculée sur les L derniers pixels visités comme suit:

$$\overline{\theta}_a = \frac{\sum_{i=0}^{L-1} \theta_{a-i}}{L},$$

où θ_{a-i} est la direction de ligne du $i^{\text{ième}}$ pixel visité avant le pixel a.

Dans le cas où aucun des trois voisins n'est un pixel de ligne, nous vérifions dans un voisinage plus éloigné dans la direction de la ligne, c'est-à-dire que lorsque la direction du vecteur entre le parent et le pixel est dans l'intervalle $[\theta_p - \pi/8, \theta_p + \pi/8]$, nous considérons ce pixel comme un enfant. Dans ce cas, le pixel suivant est l'enfant minimisant:

$$(1 - \omega)||pt_c - pt_p||_2 + \omega|\theta_c - \overline{\theta}_p| \tag{3.11}$$

où pt_c et pt_p sont les positions de l'enfant et du parent respectivement et ω représente le compromis entre la direction et la distance. Nous reprennons au début avec l'enfant

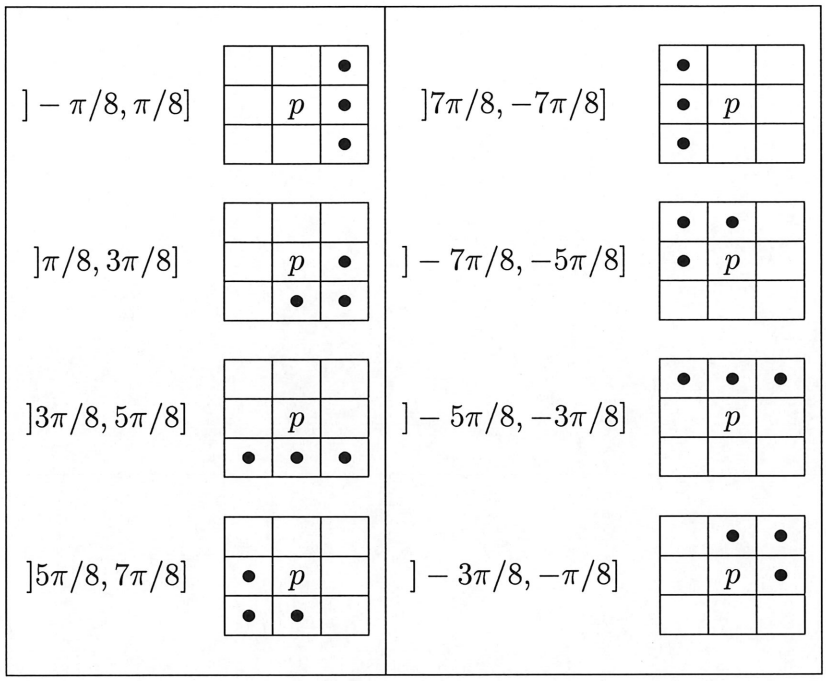


Tableau 3.1 – *Intervalles des directions de ligne et voisins considérés. p est le pixel parent et les • sont les voisins à vérifier.*

retenu. Quand un parent n'a aucun enfant, le suivi de ligne est arrêté. Si la nouvelle route comporte moins de pixel qu'un certain seuil, elle est abandonnée.

Soit une jonction située sur une ligne près du réseau existant, pour chaque pixel de ligne dans un voisinage 5×5, nous pouvons résumer l'algorithme de suivi de ligne comme suit:

1. Étiquetter le pixel comme parent.
2. Déterminer les trois voisins dans la direction de la ligne du parent (voir la table 3.1). S'ils sont des pixels de ligne, les étiquetter comme enfants.
3. S'il y a au moins un enfant:
   (a) Étiquetter l'enfant minimisant l'équation 3.10 comme parent.
   (b) Retourner à l'étape 2.

4. S'il n'y a aucun enfant:

   (a) Déterminer les enfant éloignés dans la direction de la ligne.

   (b) S'il y a au moins un enfant:

      i. Étiquetter l'enfant minimisant l'équation 3.11 comme parent.

      ii. Retourner à l'étape 2.

5. Si la longueur de la nouvelle ligne est plus petite que S, alors l'abandonner.

Dans la figure 3.11, nous avons le résultat du suivi de ligne ($L = 10, S = 7, \omega = 0.9$).

Figure 3.11 – *Hypothèses des nouvelles routes générées par le suivi de ligne ($L = 10, S = 7$ and $\omega = 0.9$).*

3.6 Résultats expérimentaux

Dans cette section, nous présentons des résultats obtenus par notre méthode. Nous avons implanté cette méthode en C++ sur une plateforme Sun et en Visual C++ sur une plateforme Windows. L'ortho-image digitale géoréfencée utilisée a une résolution spatiale de 1.5m. C'est une image de la région d'Edmonton, Alberta. Elle a été générée en numérisant une photographie aérienne d'échelle 1 : 60000 prise en 1995. Cette numérisation a été

effectuée à 1000 ppp. Les images sur lesquelles a été testé la méthode sont des sections de cette ortho-image (11334 $\times$ 10166 pixels) que nous avons rééchantillonée afin de réduire la résolution spatiale par un facteur de deux. La figure 3.16a est l'image originale avec les segments de route de la base de données et les jonctions de lignes (le seuil pour la détection des lignes est 20 et celui pour les jonctions est 0.15) mises en correspondance avec les extrémités de segments de route ($N_m = 9$). La figure 3.16b est la plausibilité de ligne claire ($\sigma = 2.5$) et la figure 3.16c est le résultat final de l'étape de correction ($\alpha = 100$ et $\beta = 0$). Comme nous pouvons voir, la partie en bas à droite a été corrigée de façon satisfaisante. Nous voyons à partir de la partie gauche du segment horizontal (en bas à gauche: celui qui passe par dessus la rivière), que la rigidité du contour actif est très utile. Malgré le fait que le détecteur de lignes ne donne pas un résultat droit partout dans ce segment, nous obtenons un résultat presque droit lors de la correction. Dans le prolongement de cette même route, nous pouvons voir que la route est juste à côté d'une ligne foncée sur un fond clair. Cela peut être un ombrage ou autre. Le contour est donc attiré par le côté de la route au lieu du milieu. Cela est un problème fréquent dans les régions urbaines. Nous observons un problème aux jonctions de routes, c'est-à-dire que la perpendicularité des routes à ces endroits n'est pas conservée. Ce problème est dû au fait qu'aux points de jonctions de lignes, la plausibilité de ligne est plus faible qu'au centre d'une ligne (sans jonction). Le contour actif est donc attiré par les régions de ligne juste à côte des jonctions. Ce problème n'a pas été investigué, mais une solution possible serait d'utiliser des contours actifs avec extrémités fixes dans les cas où il existe une jonction mise en correspondance avec l'extrémité du segment de route. Le déplacement moyen des pixels de segments de route est 2.26329 et la variance de ce déplacement est 1.46899. Nous avons eu un nombre moyen d'itérations de 47 par pixel dans l'étape de correction, pour un total de 1957 pixels traités.

Pour comparer, nous n'avons pas utilisé les informations de jonction comme dans la

méthode de Klang [24]. Nous prenons le point dans un voisinage (le même voisinage que dans la figure 3.16a), qui a la corrélation maximale donnée par l'équation 3.2. Les autres paramètres sont les mêmes. La figure 3.17 présente les résultats de la correction. Dans cette image, les résultats sont subjectivement à peu près les mêmes. Cependant, qualitativement, la variance du déplacement est plus grande qu'avec notre approche. Le déplacement moyen des pixels de segment de route est 2.3549 et la variance de ce déplacement est 2.27092. Le nombre moyen d'itérations est de 39 par pixel. Nous voyons que cela a pris un peu moins de temps à éxécuter qu'avec notre approche.

La figure 3.18 présente les résultats sur une deuxième section de l'image. La figure 3.18a est l'image originale superposée avec les segments de route de la base de données et les jonctions de lignes. Nous remarquons que la route verticale est déplacée légèrement vers la gauche. Le paramètre d'échelle σ est 2.5 et les seuils pour la détection de ligne et pour la détection de jonction sont 20 et 0.09 respectivement. La figure 3.18b est la plausibilité de ligne claire. La figure 3.18c est le résultat final de l'étape de correction ($\alpha = 100$ et $\beta = 0$). Nous pouvons voir que la route verticale est mieux localisée. Nous avons un déplacement moyen de 1.68 pixels et la variance est de 1.32. Cela a pris 22 itérations par pixel pour un total de 908 pixels.

La figure 3.19 présente les résultats obtenus à chaque étape de notre algorithme de correction sur une autre section d'image. La figure 3.19a est l'image originale avec les vecteurs de la base de données. Visuellement, nous pouvons remarquer qu'à plusieurs endroits, les vecteurs ne se superposent pas avec les routes dans l'image. La figure 3.19b est la plausibilité de ligne. La figure 3.19c est le résultat de la détection de lignes, avec une échelle (σ) de 2 et un seuil de 20. La figure 3.19d présente les jonctions de lignes (le seuil est 0.1) et les extrémités des segments de route, représentées par des carrés noirs. Les carrés blanc sont les jonctions correspondantes aux extrémités ($N_m = 13$). Nous remarquons qu'excepté pour les culs-de-sac, il n'y a qu'une extrémité qui n'a pas

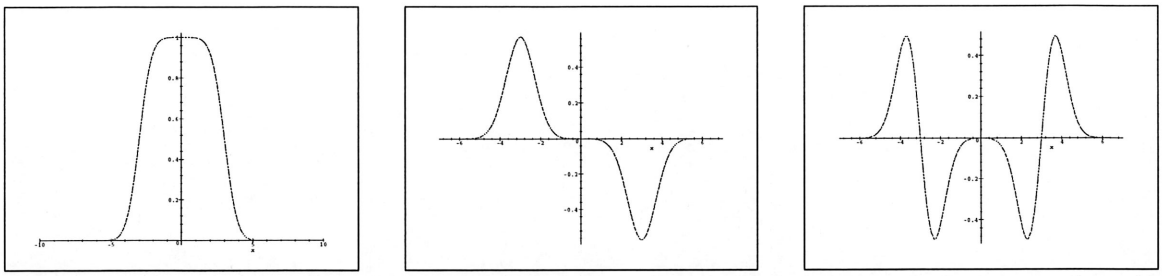
de jonction correspondante (en haut de la courbe centrale), car il n'y en a aucune dans le voisinage de l'extrémité. Si la jonction la plus proche avait été mise en correspondance, le segment central aurait été déplacé vers la droite, donc, en s'éloignant de la route. La figure 3.19e montre l'initialisation après avoir relocalisé les segments de routes. La figure 3.19f présente la position finale du contour actif ($\alpha = 100$ and $\beta = 4$). Nous remarquons que la position des contours finaux est plus près des centres de ligne que les vecteurs originaux. Le déplacement moyen des segments de route est 2.75629 pixels et la variance de ce déplacement est 4.09446. Le nombre d'itérations moyen est de 31 par pixel dans l'étape de contours actifs pour un total de 641 pixels traités.

Les résultats sans la détection de jonctions (approche de Klang avec les même paramètres que dans la figure 3.19) sont présentés dans la figure 3.20. Le déplacement moyen des pixels des segments de route est de 3.62806 et la variance de ce déplacement est de 4.95488. Le nombre moyen d'itérations est de 34 par pixel. Les routes sont mieux localisées avec notre approche.

Nous présentons maintenant les résultats concernant la génération d'hypothèses pour les nouvelles routes. Ces hypothèses ne sont pas nécéssairement des routes. La figure 3.18d est le résultat final sur la figure 3.18a. La figure 3.21 présente la détection des nouvelles routes sur la figure 3.19a. La figure 3.21a présente les jonctions de lignes près du réseau existant ($N = 7$) et la figure 3.21b est le résultat final ($L = 10, S = 7$ et $\omega = 0.9$).

Dans ce qui suit, nous discutons brièvement de l'influence des différents paramètres. Nous débutons par le paramètre d'échelle du détecteur de lignes. Le paramètre d'échelle doit être sélectionné avec attention en tenant compte de la largeur de la route à détecter [31]. S'il est trop bas, la ligne est vue comme deux contours de type marche, séparés au lieu d'une ligne. Le détecteur de lignes ne détecte pas les contours de type marche (figure 3.12). De plus, le bruit aura une plus grande influence sur les résultats. D'autre part, si l'échelle est trop grande, le lissage peut aplatir la ligne et la plausibilité sera trop basse.

Pour une discussion sur les paramètres du détecteur de jonctions de lignes voir [12].

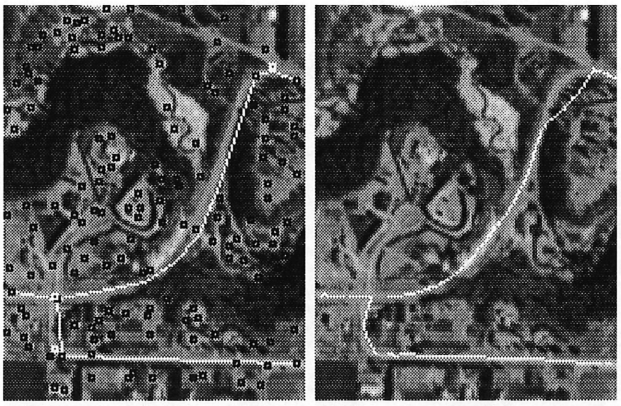


(a) (b) (c)

Figure 3.12 – *(a) Profil d'une ligne lissée avec un petit σ. (b) Première dérivée. (c) Deuxième dérivée.*

Dans l'étape de mise en correspondance, nous avons un paramètre déterminant la distance minimale permise entre une jonction de lignes et une extrémité de route (N_m). Ce voisinage doit être assez grand, dans le cas d'une grande imprécision, pour permettre une relocalisation du segment de route le rapprochant de la ligne. D'un autre côté, il ne devrait pas être trop grand car l'extrémité pourrait être mis en correspondance avec la mauvaise jonction et alors nous déplacerions le segment de ligne plus loin de la ligne (figure 3.13).

Concernant le processus de contours actifs, nous avons trois paramètres qui sont α, β et Δ. Premièrement, quand nous avons un paramètre de rigidité élevé (α) par rapport au paramètre d'élasticité (β), le segment de route final est très droit (figure 3.14a). Au contraire, quand α est petit comparé à β, le segment final a tendance à suivre les petites variations de la fonction de plausibilité de ligne (figures 3.14b et 3.14c). La principale différence entre la figure 3.14b et la figure 3.14c est le nombre d'itérations qui est plus grand pour 3.14c (53.6 itérations par pixel) que pour 3.14b (35.2 itérations par pixel). En moyenne, nous observons que lorsque $\alpha = 0$ et β augmente, le nombre d'itérations augmente aussi. Nous avons avantage à essayer de réduire le nombre d'itérations par pixel



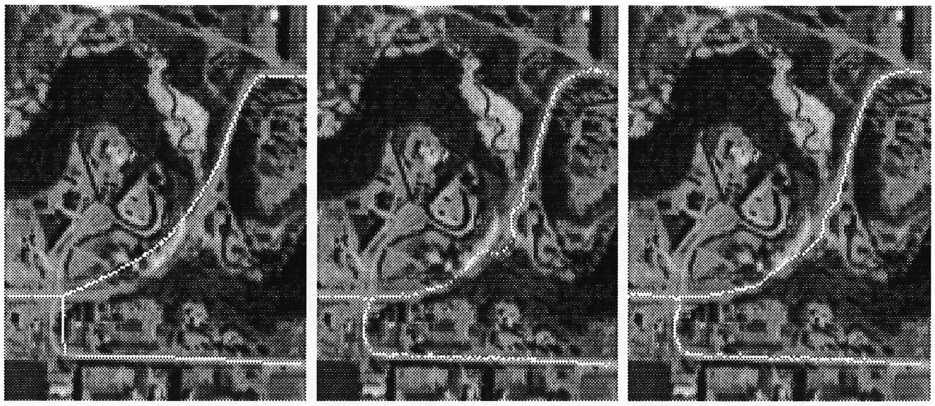
(a) (b)

Figure 3.13 – *Paramètres de mise en correspondance a) Mise en correspondance avec* $N_m = 15$. *b) Résultat final de la correction avec* $\alpha = 100, \beta = 4, \Delta = 1$.

puisque celui-ci augmente considérablement le temps de calcul. La grosseur de l'image a peu d'influence dans le temps de calcul, c'est plutôt le nombre de routes et leur longueur qui modifie ce temps. Pour les routes, il est préférable que la rigidité soit plus importante que l'élasticité, mais le contour doit être capable de suivre les routes à faible courbure.

La progression moyenne initiale (Δ) du contour influence le nombre d'itérations requises pour optimiser l'énergie. Si le Δ initial augmente, alors le nombre d'itérations diminue. Par exemple, pour obtenir le résultat montré dans la figure 3.14d, cela a pris 60.3 itérations par pixel et pour obtenir le résultat montré dans la figure 3.9, cela a pris 54.6 itérations par pixel. Cependant, au dessus d'une certaine valeur, le contour peut se retrouver en dehors de la zone d'attraction et alors évoluer dans une mauvaise direction (figure 3.14e).

Pour l'étape de la détection de nouvelles routes, nous avons trois paramètres: L, S et ω. L est le nombre de pixels sur lesquels la direction moyenne est calculée. En gros, cela influence la direction de recherche dans le cas où il n'y a pas de voisins immédiats et où nous devons effectuer une recherche dans un voisinage plus éloigné (étape 4 de



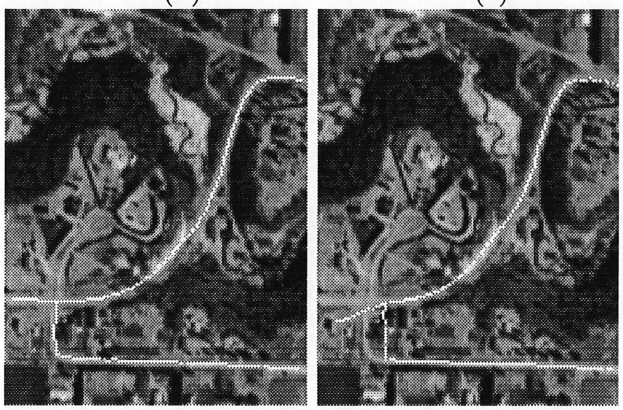


Figure 3.14 – *Paramètres du contour actif a) Résultats avec* $\alpha = 500, \beta = 4, \Delta = 1$. *b) Résultats avec* $\alpha = 0, \beta = 4, \Delta = 1$. *c) Résultats avec* $\alpha = 0, \beta = 100, \Delta = 1$. *d) Résultats avec* $\alpha = 100, \beta = 4, \Delta = 0.1$. *e) Résultats avec* $\alpha = 100, \beta = 4, \Delta = 3$.

l'algorithme de suivi de ligne). Par exemple, pour la figure 3.15a, nous avons utilisé $L = 5$, nous pouvons voir que quelques lignes sont manquantes en comparaison avec la figure 3.21b ($L = 10$). S est la longueur minimale d'une ligne retenue (figures 3.15b et 3.15c). Dans 3.15c, où $S = 15$, nous voyons que certaines lignes ont été abandonnées par rapport à 3.15b, où $S = 3$. ω représente le compromis entre la direction moyenne et la distance entre le parent et un voisin éloigné (étape 4 de l'algorithme de suivi de ligne). Par exemple, dans la figure 3.15d nous ne prennons pas la distance en compte ($\omega = 1$)

et cela change la façon dont certaines lignes sont connectées entres elles.

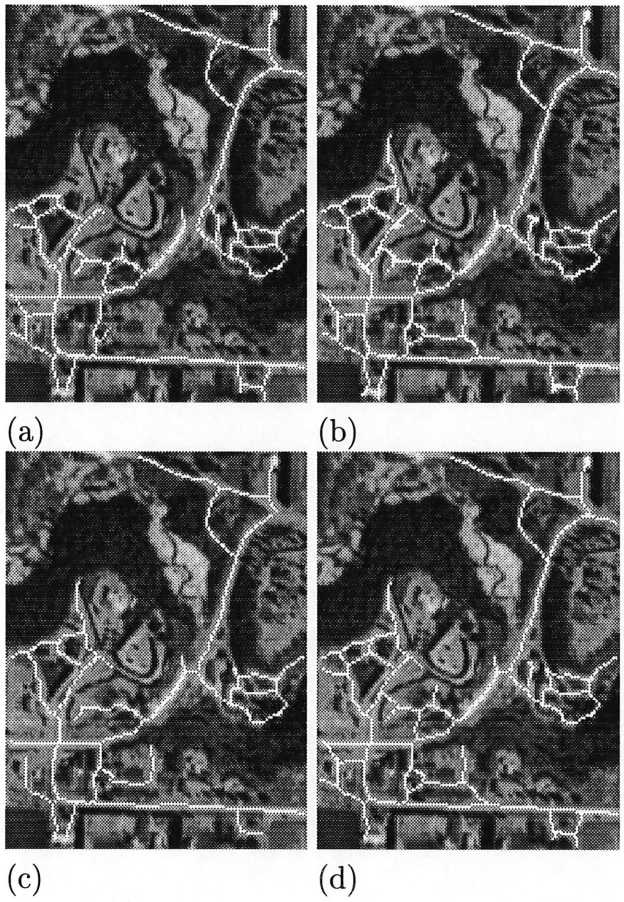
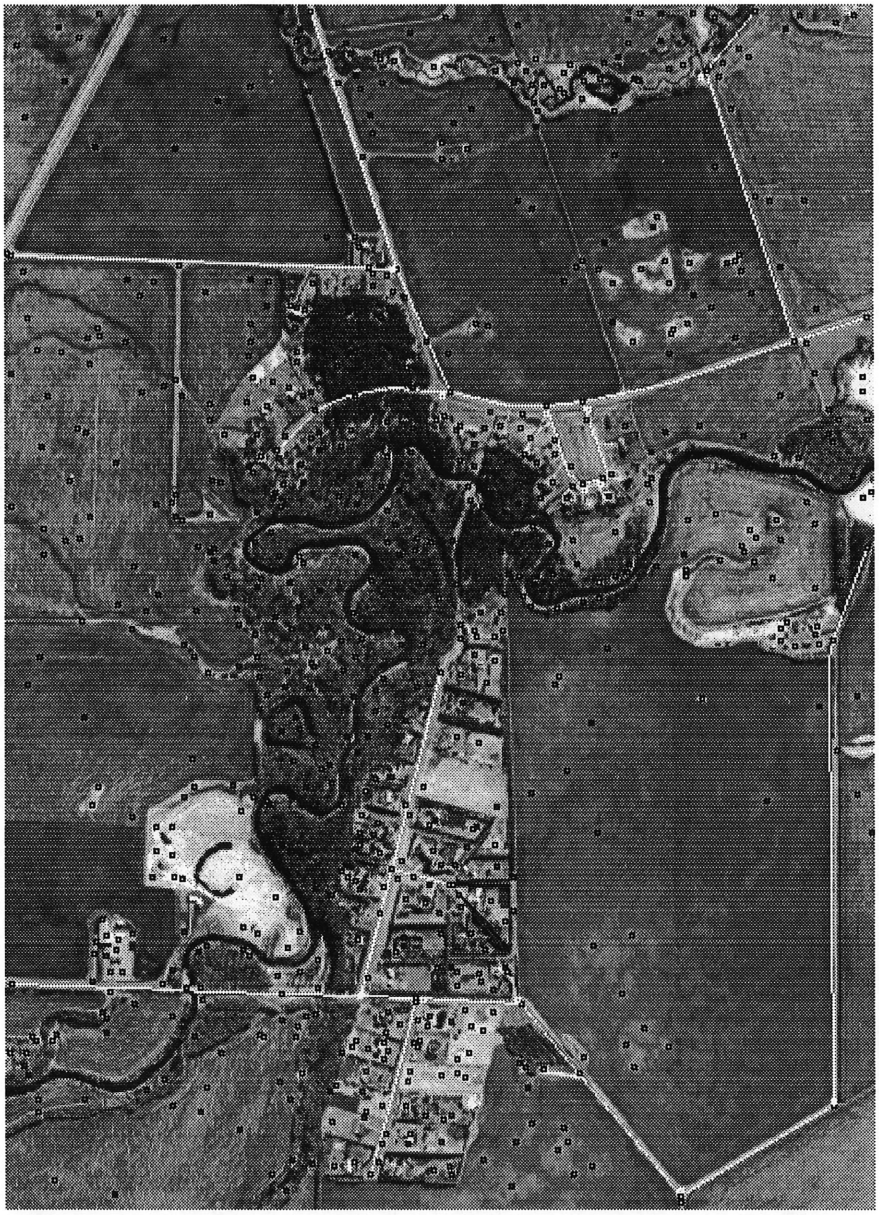


Figure 3.15 – *Paramètres du suivi de route. a) Hypothèses pour les nouvelles routes avec* $L = 5, S = 7$ *et* $\omega = 0.9$. *b) Hypothèses pour les nouvelles routes avec* $L = 10, S = 3$ *et* $\omega = 0.9$. *c) Hypothèses pour les nouvelles routes avec* $L = 10, S = 15$ *et* $\omega = 0.9$. *d) Hypothèses pour les nouvelles routes avec* $L = 10, S = 7$ *et* $\omega = 1$.



(a)

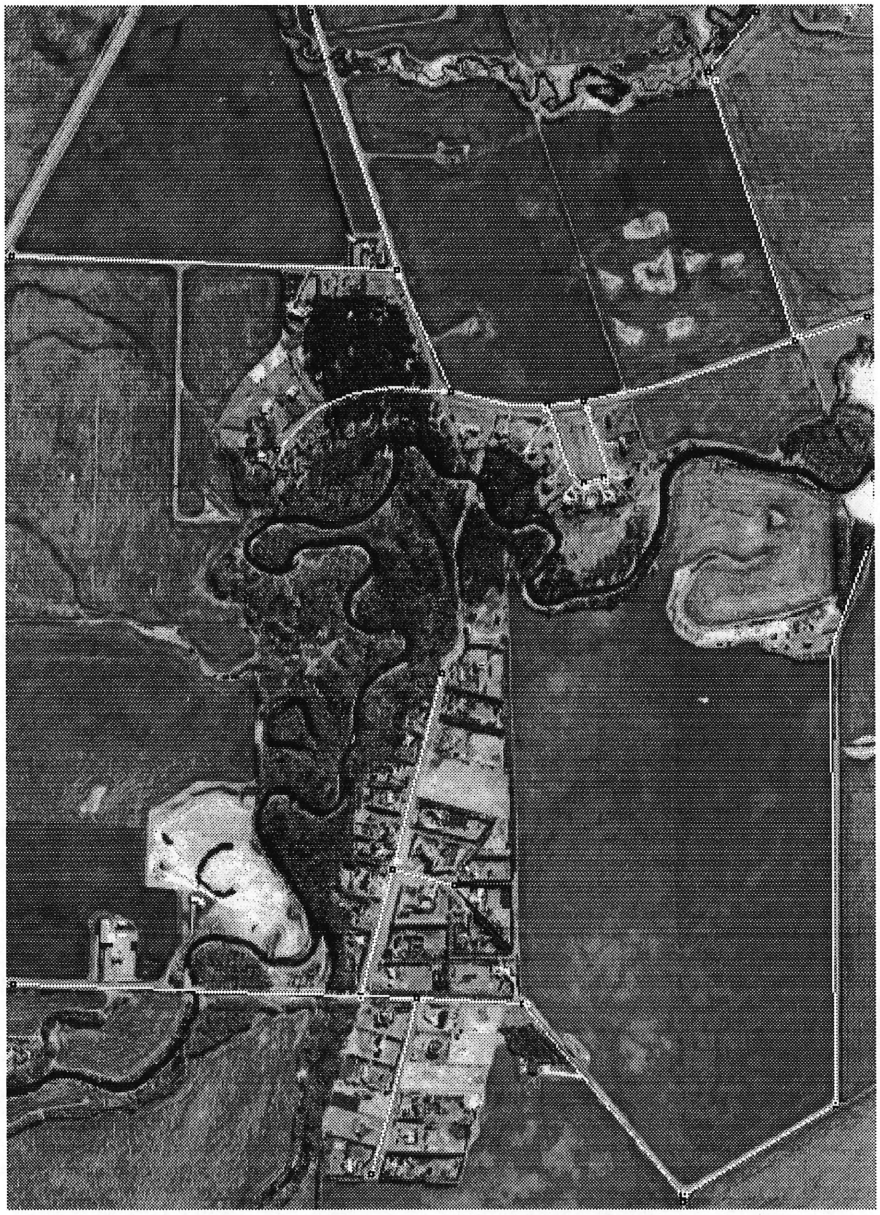
Figure 3.16 – *Image 1 a) Superposition de l'image originale (*600×433*) avec les vecteurs de la base de données, les jonctions (les carrés blancs sont les jonctions correspondantes aux extrémités de route), et les extrémités de route (carrés noirs). Les seuils pour la détection de lignes et des jonctions sont* 20 *et* 0.15 *respectivement et le voisinage pour la mise en correspondance est* 9×9.

(b)

Figure 3.16 (suite) : *b) Plausibilité de ligne claire ($\sigma = 2.5$).*

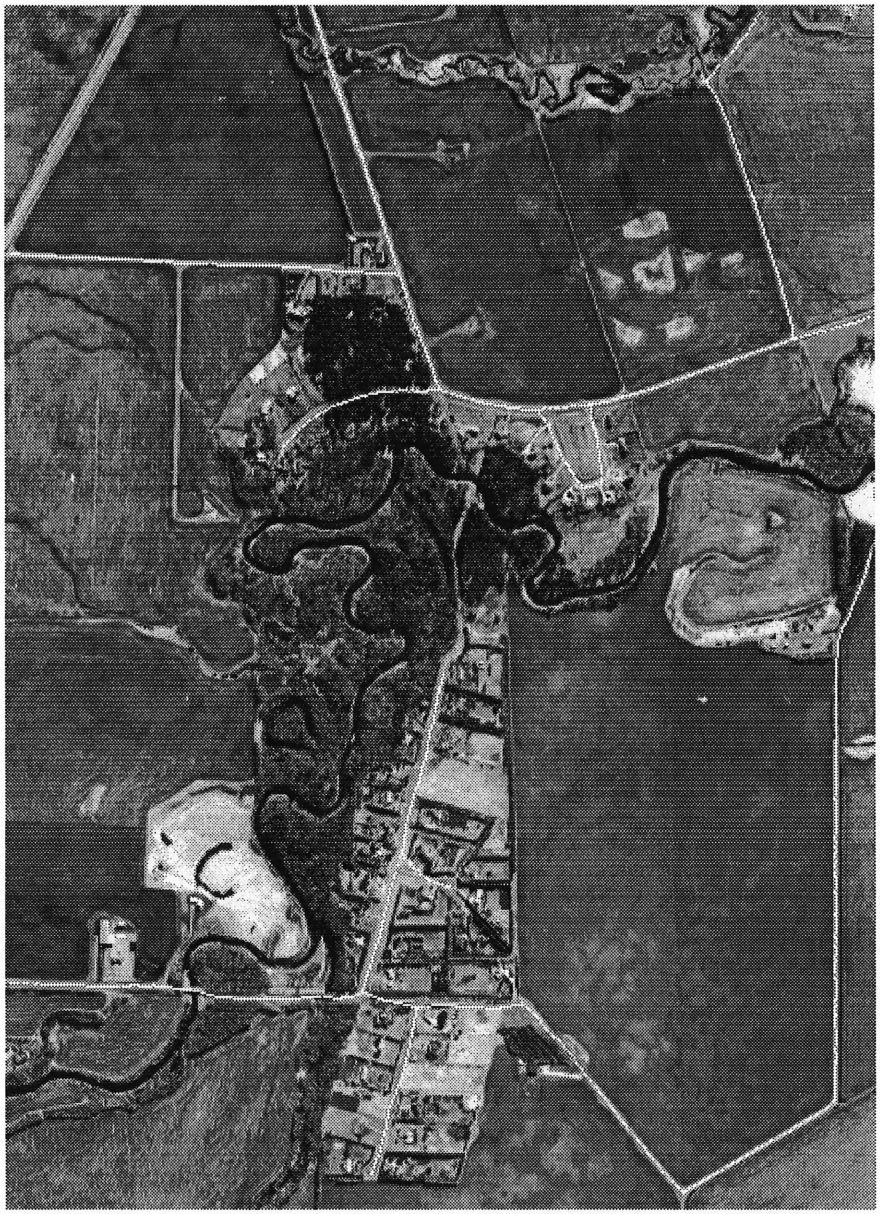
(c)

Figure 3.16 (suite) : *c) Résultat final de la correction ($\alpha = 100$ et $\beta = 0$).*



(a)

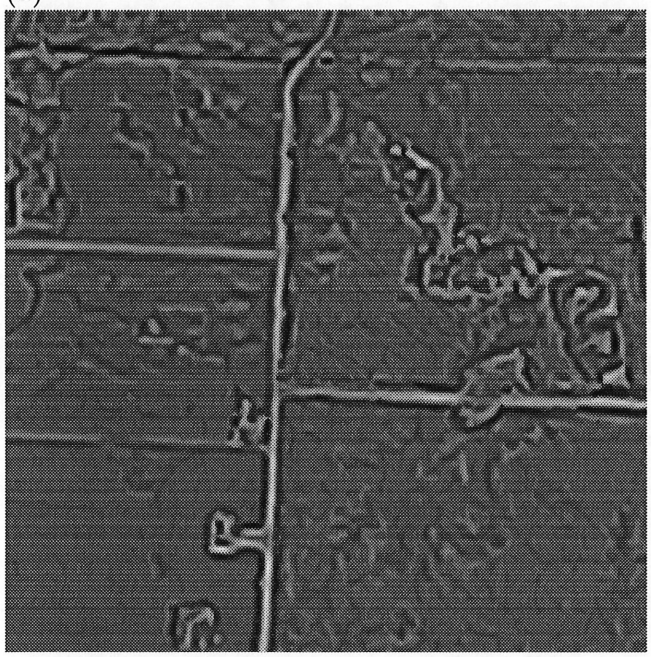
Figure 3.17 – *Méthode de Klang sur l'image 1. a) Superposition de l'image originale (*600×433*) avec les vecteurs de la base de données, les points de corrélation maximale (carrés blancs) et les extrémités des segments de route. Le seuil pour la détection de lignes est* 20 *et le voisinage pour la mise en correspondance est* 9×9.



(b)

Figure 3.17 (suite) : *c) Résultat final de la correction ($\alpha = 100$ et $\beta = 0$).*

(a)



(b)

Figure 3.18 – *Image 2 a) Image originale* (455×455) *avec les vecteurs de la base de données (lignes blanches), les jonctions de lignes (carrés noirs). b) Plausibilité de ligne claire* ($\sigma = 2.5$).

(c)

(d)

Figure 3.18 (suite) : *c) Résultat final de la correction ($\alpha = 100$ et $\beta = 0$). d) Hypothèses pour les nouvelles routes.*

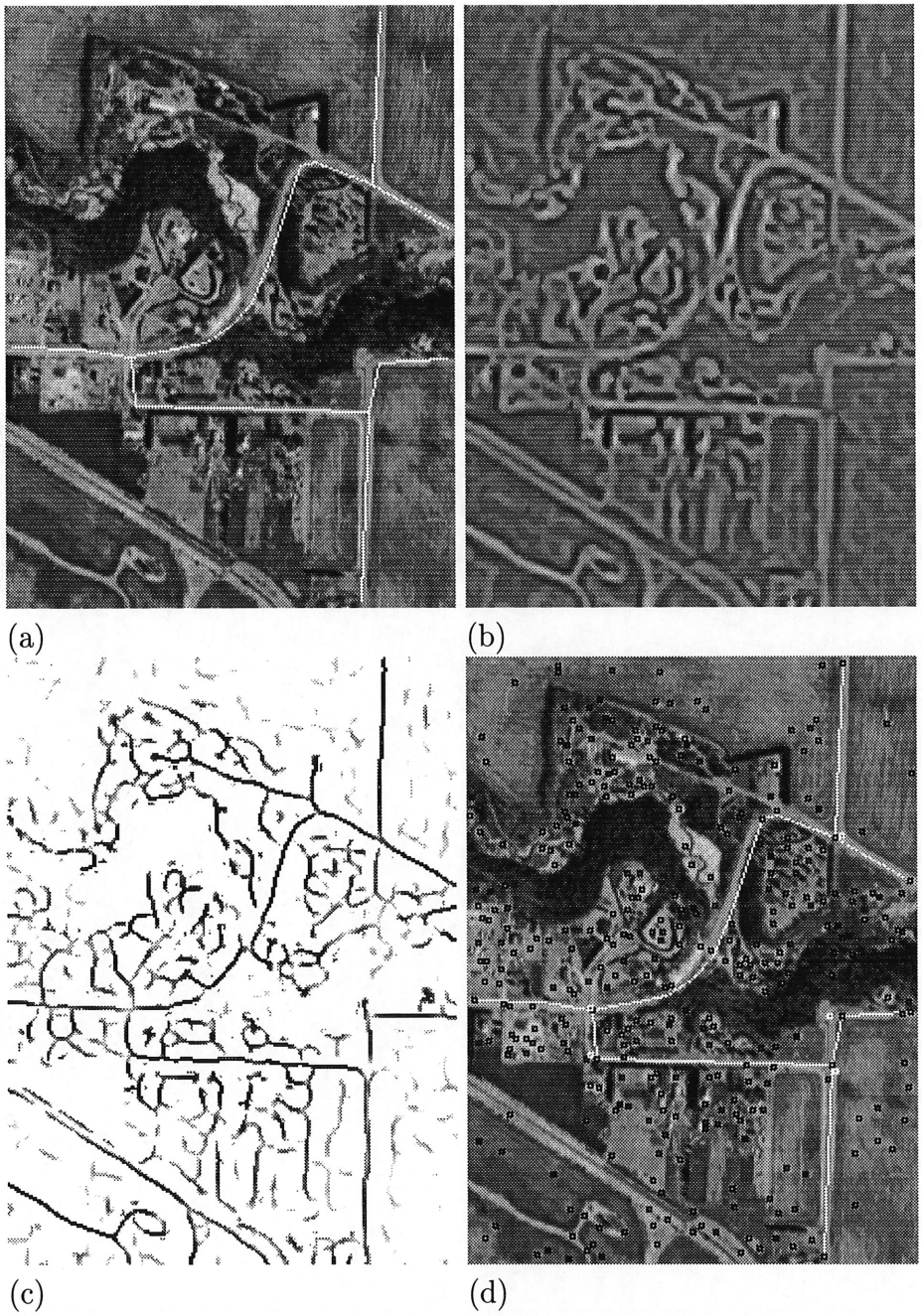
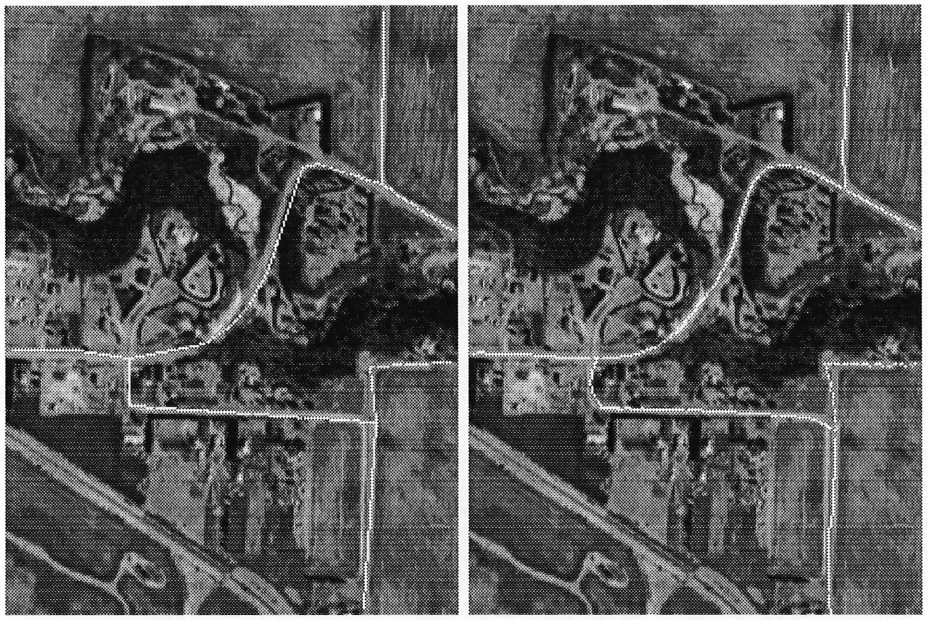
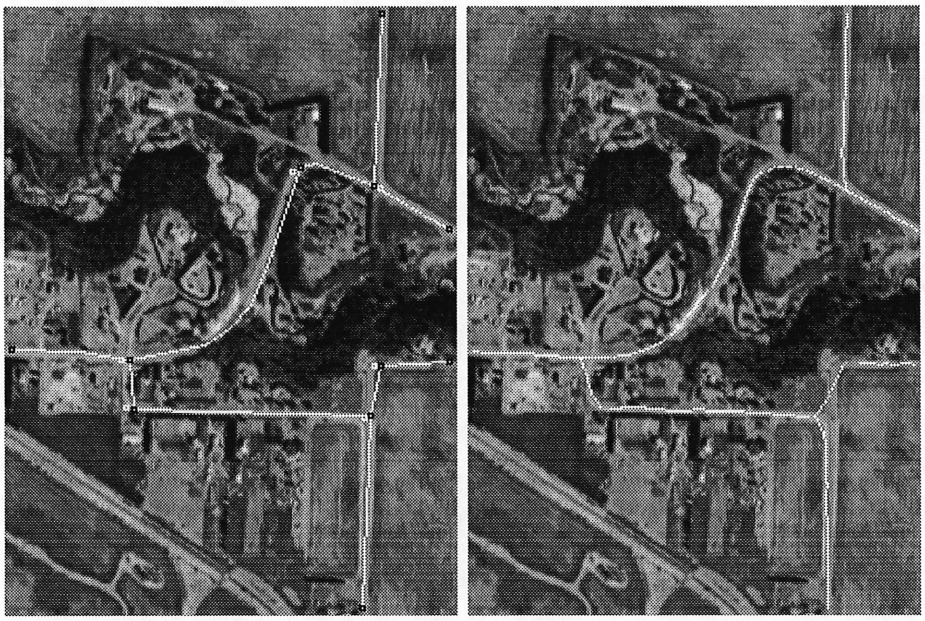


Figure 3.19 – *Image 3 a) Image originale (*217×295*) avec les vecteurs de la base de données. b) Plausibilité de ligne claire (*$\sigma = 2$*). c) Lignes détectées (le seuil est* 20*). d) Jonctions de lignes (le seuil est* 0.1*) et extrémités des segments de route (carrés noirs). Les carrés blancs sont les jonctions mises en correspondance avec les extrémités (le voisinage* N_m *est* 13×13*).*



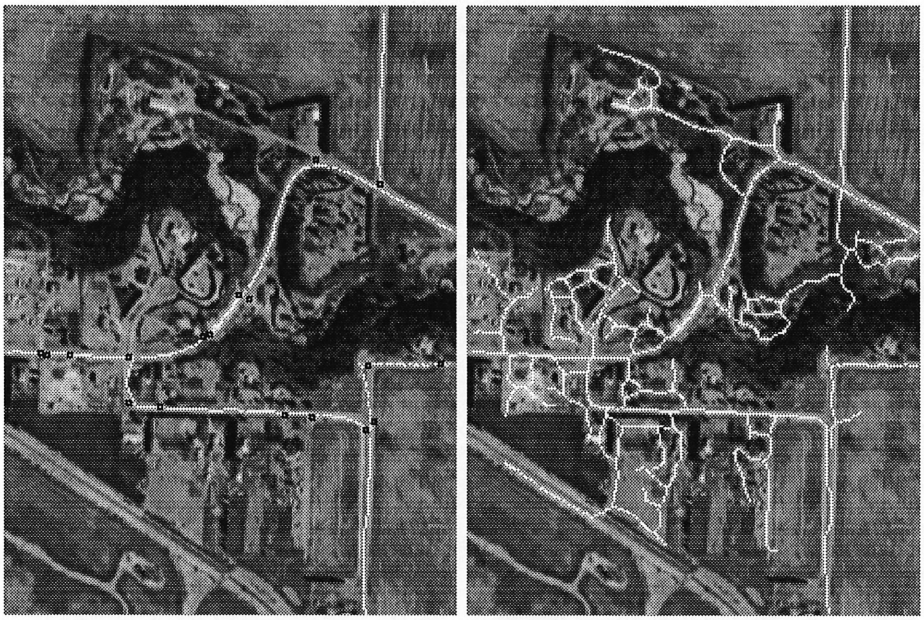
(e) (f)

Figure 3.19 (suite) : *e) Initialisation des contours actifs après la relocalisation. f) Résultat final des contours actifs ($\alpha = 100$ et $\beta = 4$).*



(a) (b)

Figure 3.20 – *Résultats sur l'image 3 sans les jonctions de lignes (Klang). a) Mise en correspondance des segments de route avec l'image. Les carrés blancs sont les points mis en correspondance. b) Résultat final* ($\alpha = 100$ *et* $\beta = 4$).



(a) (b)

Figure 3.21 – *Génération d'hypothèses pour les nouvelles routes. a) Jonctions de lignes près du réseau routier corrigé. b) Résultat ($L = 10, S = 7$ et $\omega = 0.9$).*

3.7 Conclusion et perspectives

Nous avons présenté notre approche pour la mise à jour automatique des cartes routières à partir d'images à haute résolution. La mise à jour des informations de route comporte deux problèmes, c'est-à-dire la correction de l'information existante et l'addition des nouvelles routes. Puisque les routes sont des éléments longs et minces dans les images à haute résolution, la méthode est basée sur la détection de lignes. L'information existante n'est pas précise, mais elle peut nous fournir une bonne indication de l'endroit des routes en comparaison avec d'autres éléments linéaires comme les rivières ou les chemins de fer. Pour la correction, nous utilisons donc une approche par contours actifs avec l'information existante comme initialisation. Pour amener l'initialisation plus près de la route, nous déplaçons les segments de route par rapport aux jonctions de l'image. Nous avons intégré un détecteur de jonctions de lignes qui a été développé dans notre groupe de recherche [12]. À notre connaissance, c'est le premier détecteur de jonctions de lignes développé. Nous avons présenté quelques résultats obtenus avec une image et une base de données fournies par Géomatique Canada. Pour la génération d'hypothèses pour les nouvelles routes, nous appliquons un suivi de ligne en partant des jonctions de l'image qui sont près du réseau existant. Nous avons montré l'intérêt d'utiliser les jonctions de lignes en comparant avec l'approche de Klang. Pour le futur, il serait intéressant d'intégrer des connaissances pour distinguer les nouvelles routes d'autres éléments linéaires comme les rivières ou les chemins de fer. Par exemple, il serait possible d'ajouter une condition de droiture pour éliminer les segments de ligne à forte courbure. Présentement, les nouvelles routes que nous trouvons incluent les routes de la base de données originale et nous n'avons pas cherché à résoudre le problème de la mise en correspondance entre les nouvelles routes et les anciennes. Une autre possibilité intéressante pour les nouvelles routes serait de permettre à l'usager de donner une initialisation et ensuite utiliser les contours actifs pour trouver une meilleure localisation. Les paramètres devraient aussi

être étudiés avec plus de détails. Une amélioration intéressante pourrait être de trouver une condition d'application pour le contour actif. En effet, nous avons vu dans la section 3.6, dans les régions plus habitées, il y a beaucoup de variations d'intensité sur les routes dûes aux ombrages, donc la plausibilité de ligne est inégale pour la route, ce qui influence beaucoup les contours actifs. Lorsque le détecteur de lignes ne donne pas un bon résultat, il pourrait être préférable de ne pas déplacer le segment de route à l'aide des contours actifs.

CONCLUSION

Dans ce mémoire, nous nous sommes intéressé à l'extraction de caractéristiques. Nous avons développé deux détecteurs de caractéristiques de bas niveau et un détecteur de caractéristiques de haut niveau.

D'abord, nous avons présenté un détecteur de contours dans les images multispectrales. Ces images sont définies comme une fonction $\mathbb{R}^2 \Rightarrow \mathbb{R}^m$ où m est le nombre de bandes. Nous avons développé un Laplacien basé sur les dérivées d'une fonction de contraste multispectrale présentée par Cumani [8]. Nous avons ensuite présenté quelques résultats de détection de contours démontrant la validité de notre Laplacien. Nous trouvons qu'il est sensible au seuil, cependant certains contours sont mieux localisés que ceux trouvés à l'aide du Laplacien régulier sur l'image d'intensité et l'utilisation de l'image d'intensité entraîne la perte de certains contours significatifs.

Ensuite, nous avons présenté un détecteur de contours dans les images de texture. Les moments d'ordre supérieur de l'histogramme sont utilisés pour décrire les régions de texture, mais ne sont pas utilisé en détection de contours. Dans un premier temps, nous proposons de calculer les moments d'ordre supérieur de façon locale et ensuite d'appliquer sur l'image du moment un détecteur de contours usuel tel le gradient ou le Laplacien. Dans un deuxième temps, nous proposons d'utiliser l'approche multispectrale afin de combiner les contours trouvés dans les différentes images des moments. Nous choisissons

cette solution car la fusion des contours est un procédé complexe. Nous avons présenté des résultats de la moyenne et de la variance combinées démontrant l'intérêt de cette méthode. Des études sur les moments d'ordre supérieur à la variance ainsi que sur l'effet de l'échelle doivent être effectuées.

En dernier lieu, nous avons présenté un détecteur de routes dans les images à basse résolution. Pour ce détecteur, nous utilisons une base de données, fournie par Géomatique Canada, contenant des informations de routes obtenues par numérisation des cartes routières. Ces informations présentent deux problèmes: 1) la localisation des routes existantes est imprécise et 2) les nouvelles routes manquent. Nous avons présenté une approche pour la mise-à-jour de ces informations qui est une variante de celle présentée dans [24]. D'abord, nous utilisons les contours actifs pour corriger les routes existantes. Nous ajoutons l'information des jonctions de lignes à cette approche dans le but de rapprocher l'initialisation fournie par la base de données de la route dans l'image. Le détecteur de jonctions de lignes est un détecteur original qui a été développé parallèlement à ce projet, entre autres, pour y être intégré. Ensuite, nous effectuons un suivi de ligne à partir du réseau existant afin de générer des hypothèses pour les nouvelles routes. Nous avons présenté des résultats sur des images fournies par Géomatique Canada, montrant l'intérêt d'ajouter les jonctions de lignes. Certaines améliorations restent à faire, par exemple l'ajout de contraintes sur les routes (droiture, largeur constante, splines, ...) pour la génération des hypothèses des nouvelles routes ou une condition d'application du contour actif.

Bibliographie

[1] N. Abbadeni, D. Ziou, and S. Wang. Recherche d'images basée sur le contenu: Représentation de la texture par le modèle autorégressif. Technical Report 216, Département de mathématiques et d'informatique, Université de Sherbrooke, Mai 1998.

[2] M. Amadasun and R. King. Textural Features Corresponding to Textural Properties. IEEE *Transactions Systems, man, and cybernetics*, 19(5):1264–1274, September 1989.

[3] M.-F. Auclair-Fortier, D. Ziou, C. Armenakis, and S. Wang. Aerial and Satellite Road Extraction Survey. Technical Report 238, Département de mathématiques et d'informatique, Université de Sherbrooke, 1999.

[4] M.-F. Auclair-Fortier, D. Ziou, C. Armenakis, and S. Wang. Nouvelles perspectives en détection de contour: Textures et images multi-spectrales. In *Vision Interface*, Trois-Rivières, Canada, May 1999.

[5] M.-O. Berger. *Les contours actifs: modélisation, comportement et convergence.* Thèse de Doctorat, Institut National Polytechnique de Lorraine, 1991.

[6] J. Canny. A Computational Approach to Edge Detection. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[7] C.-C. Chu and J. K. Aggarwal. The Integration of Image Segmentation Maps Using Region and Edge Information. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 15(12):1241–1252, December 1993.

[8] A. Cumani. Edge Detection in Multispectral Images. *CVGIP: Graphical Models and Image Processing*, 53(1):40–51, January 1991.

[9] A. Cumani, P. Grattoni, and A. Guiducci. An Edge-Based Description of Color Images. *CVGIP: Graphical Models and Image Processing*, 53(4):313–323, July 1991.

[10] C. J. Delcroix and M. A. Abidi. Fusion of Edge Maps in Color Images. *Proc. SPIE-Int. Soc. Opt. Eng.*, 1001:545–554, 1988.

[11] R. Deriche and G. Giraudon. A Computationnal Approach for Corner and Vertex Detection. *The International Journal of Computer Vision*, 10(2):101–124, 1993.

[12] F. Deschênes and D. Ziou. Detection of Line Junctions and Line Terminations Using Curvilinear Features. Technical Report 235, Département de mathématiques et d'informatique, Université de Sherbrooke, 1999.

[13] S. DiZenzo. A Note on the Gradient of a Multi-Image. *Computer Vision, Graphics and Image Processing*, 33:116–125, 1986.

[14] P. M. Djurie and J.-K. Fwu. On the Detection of Edges in Vector Images. IEEE *Transactions on Image Processing*, 6(11):1595–1601, 1997.

[15] C. Drewniok. Multi-Spectral Edge Detection: Some Experiments on Data from Landsat-TM. *Int. Journal of Remote Sensing*, 15(18):3743–3765, 1994.

[16] C. Drewniok and L. Dreschler-Fischer. A Multispectral Edge Detection Scheme and its Application to Landsat Imagery. In *Proc. of the Int. Geoscience and Remote Sensing Symposium*, pages 1868–1870, 1993.

[17] D. Dunn, W. E. Higgins, and J. Wakeley. Texture Segmentation Using 2-D Gabor Elementary Functions. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 16(2):130–149, February 1994.

[18] K.-B. Eom and R. L. Kashyap. Composite Edge Detection with Random Field Models. IEEE *Transactions Systems, man, and cybernetics*, 20(1):81–93, January/February 1990.

[19] J. Foley, A. Van Dam, S. Feiner, and J. Hughes. *Computer Graphics: Principles and Practice.* The Systems Programming Series. Addison-Wesley, 1996.

[20] R. C. Gonzalez and R. E. Woods. *Digital Image Processing.* Addison-Wesley Publising Company, Reading, MA, US, 1993.

[21] D.-C. He and L. Wang. Detecting Texture Edges from Images. *Pattern Recognition*, 25(6):595–600, 1992.

[22] R. L. Kashyap and K.-B. Eom. Texture Boundary Detection Based on the Long Correlation Model. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 11(1):58–67, January 1989.

[23] A. Khotanzad and J.-Y. Chen. Unsupervised Segmentation of Textured Images by Edge Detection in Multidimensional Feature. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 11(4):414–421, April 1989.

[24] Dan Klang. Automatic Detection of Changes in Road Databases Using Satellite Imagery. In *Proceedings of International Archives of Photogrammetry and Remote Sensing*, volume 32, pages 293–298, 1998.

[25] H.-C. Lee and D. R. Cok. Detecting Boundaries in a Vector Field. IEEE *Transactions on Signal Processing*, 39(5):1181–1194, 1991.

[26] B.-C. Li and J. Shen. Fast Computation of Moment Invariants. *Pattern Recognition*, 24(8):807–813, 1991.

[27] B. S. Manjunath and R. Chellappa. A Unified Approach to Boundary Perception: Edges, Textures, and Illusory Contours. IEEE *Transactions on Neural Networks*, 4(1):96–107, January 1993.

[28] C. L. Novak and S. A. Shafer. Color Edge Detection. In *Proc. DARPA-Image Understanding Workshop*, pages 35–37, Los Angeles, CA, February 1987. Morgan Kaufmann Publ. Inc., Los Altos, CA.

[29] J. Scharcanski and A. N. Venetsanopoulos. Edge Detection of Color Images Using Directional Operators. IEEE *Transactions on Circuits and Systems for Video Technology*, 7(2):397–401, April 1997.

[30] P. De Souza. Edge Detection Using Sliding Statistical Tests. *Computer Vision, Graphics and Image Processing*, 23:1–14, 1983.

[31] C. Steger. An Unbiased Detector of Curvilinear Structures. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 20(2):113–125, February 1998.

[32] S. Tabbone. Detecting Junction Using Laplacian of Gaussian Operator. In *Proceedings of the* 12^{th} *International Conference on Pattern Recognition*, pages 52–56, 1994.

[33] W. B. Thompson. Textural Boundary Analysis. IEEE *Transactions on Computer*, 26:272–276, march 1977.

[34] M. Tuceryan and A. K. Jain. Texture Analysis. *Handbook of Pattern Recogition and Computer Vision*, pages 235–276, 1993.

[35] M. Turcotte, S. Wang, and J. Vaillancourt. Extraction et sélection de caractéristiques de texture pour la classification d'images. Technical Report 231, Département de mathématiques et d'informatique, Université de Sherbrooke, Mars 1999.

[36] S. R. Yhann and T. Y. Young. Boundary Localization in Texture Segmentation. IEEE *Transactions on Image Processing*, 4(6):849–856, June 1995.

[37] D. Ziou and S. Tabbone. Edge Detection Techniques - An Overview. *Pattern Recognition and Image Analysis*, 8(4):537–559, 1998.